Semi-autonomous brachiating robot for teleoperated steep terrain tree harvesting

Christopher Vincent Meaclem

Department of Mechanical Engineering
University of Canterbury

A thesis submitted in partial fulfilment of the requirements for the degree of
Doctor of Philosophy

March 2016

To Lord Jesus be the glory...

Acknowledgements

My work over these years, culminating in this thesis, would not have been possible if not for the steadfast love and support from my fiancée. Rayna, this work is as much yours as it is mine - a statement of what we can achieve together. You have been my motivation, encourager and reinforcement. Thanksgiving must be offered to my parents. Robin and Johanna, you have equipped me at every stage in my journey to purse my dreams. My caring in-laws, Vijay and Rachael, it is a blessing to have had your wisdom around me throughout this. The prayers and advice were always appreciated. Thomas, my brother, my friend, sincere thanks for standing with me, listening and showing what it means to be a brother.

Academically this work would not have started had it not been for my primary supervisor, Prof. XiaoQi Chen, believing in me. Dr. Richard Parker, I thank you for always being the optimist, out-of-the-box thinker and sympathetic ear. Your original concepts on this project are of course why we are all here. The eye for detail, times of laughter and spiritual support only represent a small selection of the reasons I am grateful for the co-supervision of Dr. Stefanie Gutschmidt. Perhaps the lavish servings of homemade Rhubarb cake should also be mentioned. Rodney Elliott, your hand-on experience and advice throughout this project have been invaluable. The generosity of your time appreciated always and laughter of the laboratory sorely missed.

I gratefully acknowledge the financial support of Future Forests Research and specifically Keith Raymond. You have eased the financial burden and aided the project in achieving its industry milestones. Credit also to the Forest Owners Association for believing in this project.

Finally, my skilled colleagues (Alex Lippitt, Bart Milne, Hoani Bryson), academics, laboratory technicians and many others are much accredited with their assistance and advice.

One can pay back the loan of gold, but one dies forever in debt to those who are kind.
-Malayan Proverb

Abstract

In New Zealand, the forest industry is economically significant, representing the third largest export by value. Harvesting and extraction of trees pose serious health and safety issues. These issues have become more pronounced with the prevalence of forestry blocks established on steep slopes. The steep slopes ($> 20°$) preclude the operation of traditional machines such as harvesters and feller-bunchers. Without mechanization, felling is performed manually by ground crew with hand-held chainsaws. The forest industry now leads fatality and injury statistics with 42 % and 35 % respectively caused during the felling process. The purpose of this research, therefore, is to design and build a semi-autonomous tree felling robot suitable for the unique forestry conditions found in New Zealand. This is motivated by the desire to reduce the aforementioned fatality and injury statistics.

Currently available machinery for forest harvesting is not suitable for New Zealand's steep terrain. Tracked and wheeled vehicles cause significant damage to the hill side and struggle to provide sufficient traction on slopes. Vehicles with a high center of mass can roll, putting the driver in danger; a problem exacerbated by the slope inclination. Although some machines have been modified for increased stability on steep terrain, their lack of agility and continued damage to the terrain is prohibitive. Therefore, from literature of the state of the art, it is seen that there is no suitable forest harvesting machine compatible with New Zealand's terrain and the requirements of forest managers.

The proposed solution is a system using arboreal locomotion similar to that used by monkeys, resulting in a machine supported in the trees rather than on the ground. This method of traversal between trees has not been used for forestry previously and introduces a paradigm shift for forest planning. Due to the unique motion of this machine, it can traverse over terrain typically impractical for traditional ground based felling systems. In order to provide semi-autonomy and remote operation, a suite of sensors and control systems are required for this new application. A novel tree detection algorithm using laser scanning is proposed as a solution to gaining situational awareness for the environment. This enables the robot to identify neighboring trees and analyze which are suitable for traversal and are

ready to be harvested. Identification of objects such as trees, ground and other clutter are used in motion planning in order to avoid collisions. Choosing a suitable route through the forest can be a difficult problem for remote operators. Therefore a route planning algorithm is developed to find suitable routes. The planning algorithm considers the known kinematics of the machine and combines a-priori map information with gathered data to calculate routes through the entire forest to minimize a cost function. The cost function is determined by the forest manager with considerations towards fuel consumption, time and yield for example. This system allows forest managers to maximise the robot's utility. Finally, to provide tree felling capability, an autonomous felling mechanism is presented. In a novel development, force sensors are integrated into the chainsaw bar. These senors provide real-time feedback to the felling algorithm regarding the downwards force of the tree on the bar. By measuring the force, the state of the tree fall can be determined allowing correct timing to be achieved.

This document outlines the advancements to the state of the art through theoretic basis, conceptual development, methodology and results gathered in the laboratory and field. This research work is believed to represent a very significant step forward to achieving a commercially viable, steep terrain tree harvesting robot.

Publications

C. V. Meaclem, X. Chen, S. Gutschmidt, C. Hann, and R. Parker, "K-means Partitioned Space Path Planning (KPSPP) for robotic tree felling on steep terrain," *International Journal of Advanced Robotic Systems*, 2015.

A. Kolb, **C. Meaclem**, X. Chen, R. Parker, S. Gutschmidt, and B. Milne, "Tree trunk detection system using LiDAR for a semi-autonomous tree felling robot," presented at the *IEEE Conference on Industrial Electronics and Applications*, Auckland, New Zealand, 2015.

C. V. Meaclem, S. Gutschmidt, X. Chen, and R. Parker, "Kinematic and dynamic analysis of a brachiating tree-to-tree machine," presented at the *Robotics and Biomimetics (ROBIO), IEEE International Conference on* Zhuhai, China, 2015.

C. V. Meaclem, L. Shao, R. Parker, S. Gutschmidt, C. E. Hann, B. J. E. Milne, et al., "Sensor guided biped felling machine for steep terrain harvesting," presented at the *IEEE Conference on Automation Science and Engineering*, Taipei, Taiwan, 2014.

B. J. E. Milne, Y. Liu, **C. V. Meaclem**, X. Chen, C. E. Hann, and R. Parker, "Development and Testing of a Minimal Model for a Pressure Compensated Hydraulic Cylinder Control System " presented at the *Robotics and Biomimetics (ROBIO). 2015 IEEE International Conference on*, Zhuhai, China, 2015.

Submitted, pending acceptance:

C. V. Meaclem, X. Chen, S. Gutschmidt, C. Hann, and R. Parker, "LiDAR guided brachiation for a tree felling robot on steep terrain," *Transactions on Geoscience and Remote Sensing*, 2016.

C. V. Meaclem, A. Kolb, X. Chen, S. Gutschmidt, and R. Parker, "Autonomous force-sensitive cutting mechanism for a tree felling robot on steep terrain," *Robotics and Computer-Integrated Manufacturing*, 2016.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Forestry in New Zealand

New Zealand's forest industry is the country's third largest export earner, contributing 2.9% of the gross domestic product [1]. The first and second biggest export earners are also both from primary industries: "Milk powder, butter and cheese" and "meat and edible offal", respectively [2]. Natural forest accounts for 24 % [3] of the land area. Plantation forest utilizes 7 % (1.8 Mha) of land area. Principle species in these plantations are *Pinus radiata* (Pine) 89.5 %, Douglas-fir 6.2 % and Eucalyptus 1.4 %. Further to the current export importance above, the international significance of the industry is forecast to increase. In 2010, New Zealand was ranked in the top 20 of global wood suppliers and is forecast to be in the top five by 2025. Forestry export value is also forecast to increase 49.1 % between 2012 and 2015 [4]. This is due, in part, to an increase in the volume of harvestable Pine trees as plantings of the 1980s and early 1990s [5] reach commercial maturity of 25 to 30 years of age. In Fig. 1.1, captured in 2009, it can be observed that there is a significant peak in 11 to 15 year old trees. That peak will be entering harvesting age between 2019 and 2028. These forests are principally on marginal pastoral terrain such as mountainous and foothill terrain [6], know as "steep slope" areas. Steep slopes have been defined by multiple sources as between 35% to 40% [7, 8, 9, 10, 11, 12].

Increasing global and domestic demand for resources and produce has led to a large growth in use of robotic platforms in the primary industry to increase outputs. Other factors, such as a smaller rural labor force and sustainability, have also had an influence towards robotization and automation. The McKinsey Global Institute identified advanced robotics as a potentially disruptive technology, due to its capacity to completely change the status quo of existing manual and semi-mechanized paradigms [13]. Robotics in agriculture have

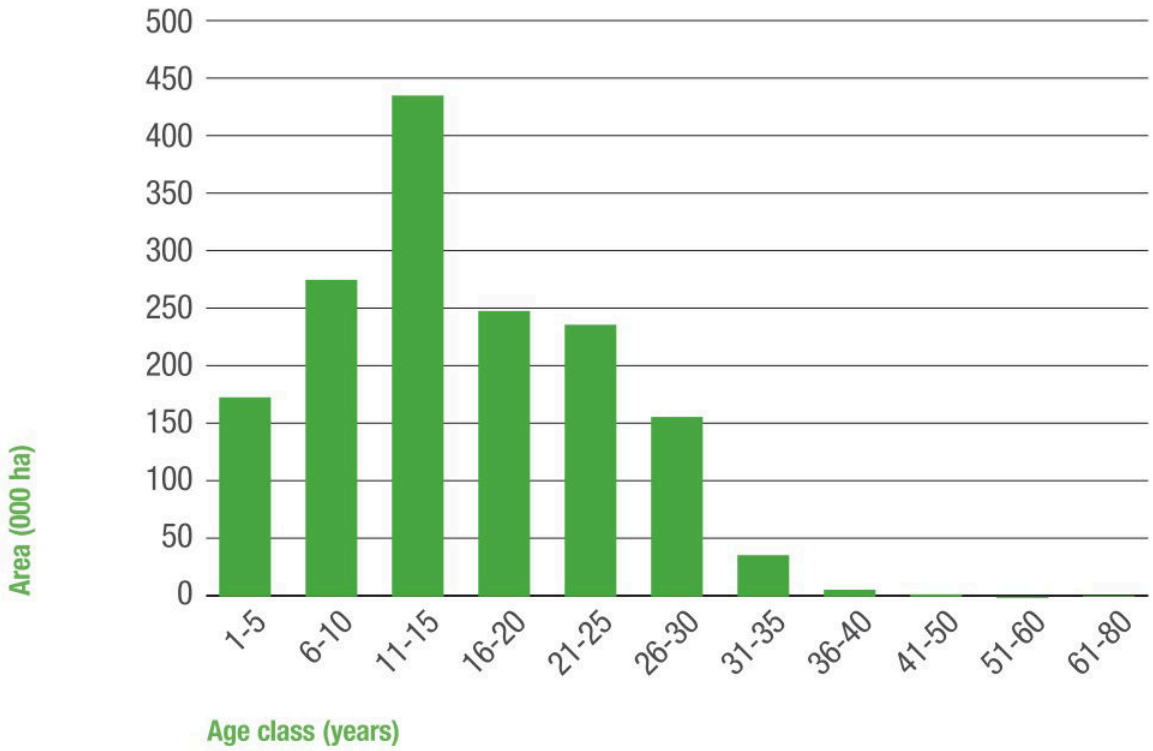


Fig. 1.1 Net stocked area of *Pinus radiata* by age class, as at 1 April 2009 [3]

the potential to decrease the vulnerability of workers in dangerous working conditions and increase the efficiency of time, materials and resources through better utilization and planning. Primary industries have already seen a wealth of promising robotic harvesting platforms but, despite success in other areas, only limited developments have occurred for the forest industry. The steep terrain of the country's plantation forests limits the utilization of machines. The forest industry is the country's most injury- and fatality-prone. The high rate of manual felling is a leading contributor to this statistic. The industry also ranks as the most deadly in the country, with a fatality rate six times higher than the average rate for all sectors. The felling process accounts for 42 % of fatalities (Fig. 1.2) and 35 % of injuries (Fig. 1.3).

An example of a commercial plantation forest is shown in Fig. 1.4. The most common harvesting method is that of "clearfelling" [16] in which an entire forest subsection known as a "stand" is felled leaving the site "clear". This felling is done manually using chainsaws or with mechanized devices such as the John Deere 909 harvester (Fig. 1.5). For steep terrain, logs are best extracted using "cable yarders" [17]. A common yarder configuration is shown in Fig. 1.6. Logs are attached to the carriage by ground crew allowing the yarder operator to drag the log, using pulleys, to a rendezvous point. This rendezvous point, or "skid", forms

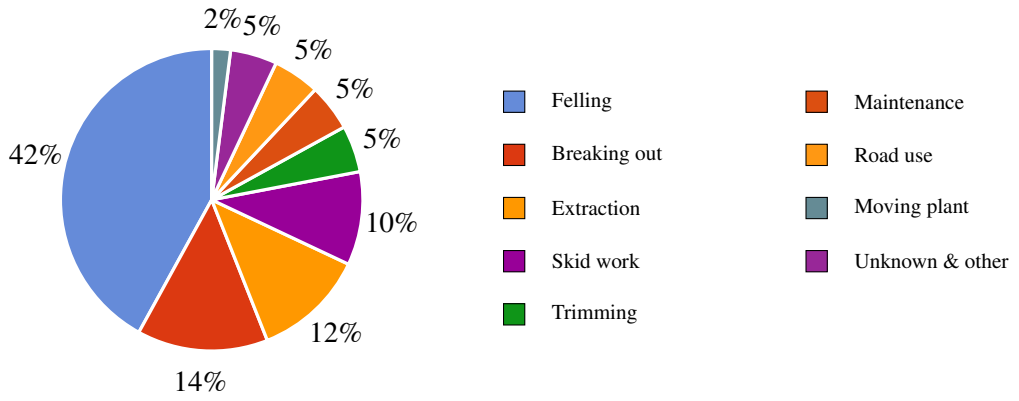


Fig. 1.2 Causes of fatalities 1988 to 2005, adapted from [14]

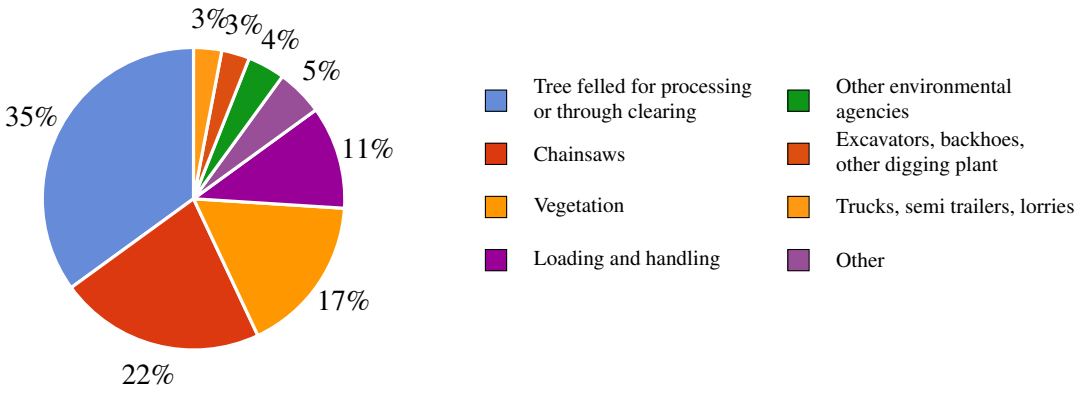


Fig. 1.3 Top ten causes of serious harm notification in forestry 2003 – 2010, adapted from [14]

Fig. 1.4 Example of a forest block using cable logging techniques near Tokoroa, New Zealand [15]

an intermediate collection point where logs are to be collected and loaded onto trucks for transport. In a forest block there can exist many active skids, allowing multiple stands to be in operation at any one time.

In recent decades' mechanization and automation significantly changed work habits in the forest industry. Excavators, for example, have been adapted for the forest industry using attachments specifically constructed for felling trees to facilitate the process. These harvesters and feller-bunchers not only increase efficiency and quality, but also workplace safety [20]. Nevertheless there are restrictions to the application of harvesters; their limited mobility on steep terrain and damage caused to soil by tracks are most salient. Due to these difficulties, mechanised felling accounts for only 23 % of New Zealand's harvest [6]. Motor-manual felling, accounting for the other 77 %, is performed by workers with light equipment such as chainsaws, levers and wedges. In manual felling, workers are directly exposed to hazards such as falling trees, falling debris and kickback of chainsaws. New Zealand statistics show that the felling component is the highest cause of deaths and injuries in the forestry sector [21, 22, 23].

The ClimbMax [24] harvester system (Fig. 1.7) represents the degree to which mechanisation of tree felling on steep terrain has been achieved in New Zealand. It is an excavator that has been modified to work on slopes up to 45° by using a winch to secure the excavator to the top of the hill for steep slope operation. The excavator has a felling head attached at the end of the arm to fell trees. In field trials [25] the machine has demonstrated an increase of 26% in volume of logs cut and bunched as well as a reduction in costs of 10% over conventional methods. The cable used to increase the mechanical stability of the machine against rolling reduces the maneuverability of the machine, whilst the tracked nature of the vehicle leads to erosion on the steep slopes. The proposed solution attempts to solve these issues by moving independently of the ground conditions since it is suspended in the trees.

Fig. 1.5 John Deere 909 tracked harvester [18]

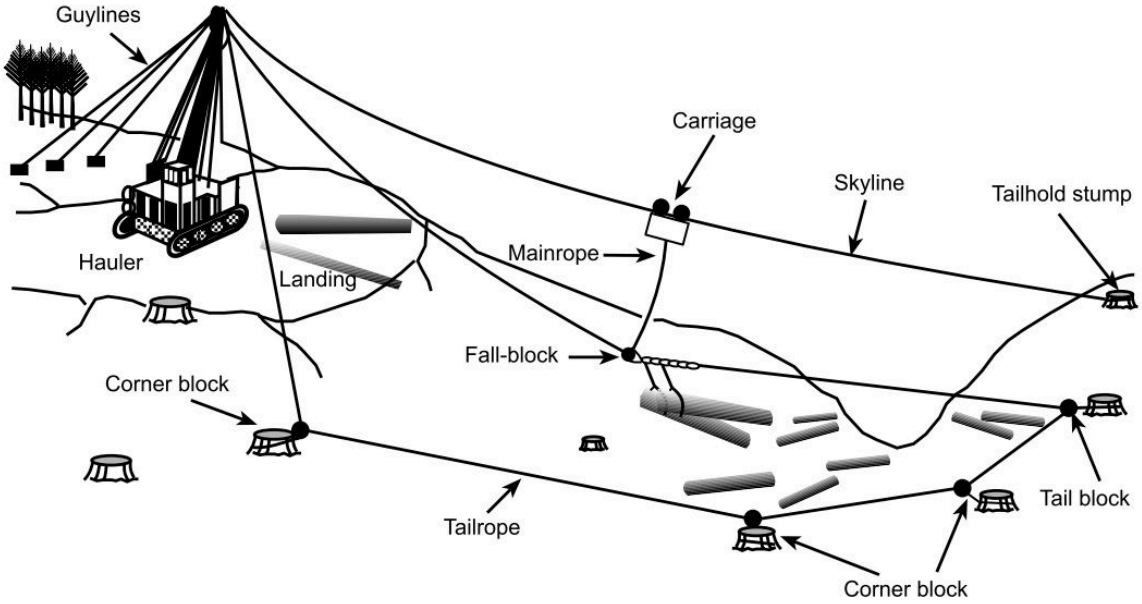


Fig. 1.6 The Northbend standing skyline configuration [19]

(a)

(b)

Fig. 1.7 A ClimbMax havesting machine [24]

1.2 Thesis layout and contributions

This thesis is organized into 8 chapters, focusing on four topics; the development of a mobility platform for forest environments (Chapters 3 and 4), route planning for harvesting forests (Chapter 5), the use of Light Detection and Ranging (LiDAR) for object detection (Chapter 6) and finally the development of a felling mechanism (Chapter 7) for the aforementioned mobility platform.

Firstly, a literature review is conducted in Chapter 2. Chapter 3 introduces the machine. Requirements for the machine's design are identified, that will allow it to operate in forests. The mechanical design is then elaborated, to demonstrate the design and construction of the novel machine. The locomotion methodology used by the machine to traverse the forest is described finally. The contributions of this chapter are to introduce the novel design of a forestry machine, as well as the new locomotion methodology not previously investigated.

Chapter 4 advances Chapter 3 by proceeding with a mechanical analysis of the machine from a kinematic and dynamic perspective. This chapter contributes to the in-depth knowledge of the mechanical properties required for control of the locomotion.

Chapter 5 addresses the proposed solution to coverage route planning in forests. This chapter contributes new knowledge to the area of route planning. Using the approach proposed in the chapter, harvesting machines and robots will be able to plan coverage routes suitable for harvesting crops such as trees. Planning for such crops was not previously available in literature.

Chapter 6 describes the solution taken to identify objects using a terrestrial LiDAR scanner. A system consisting of a sensor apparatus and detection algorithm present enhancements to the state of the art of environment detection. Using a simpler methodology than existing works, a lower error rate is achieved in the detection of trees in forest environments.

Chapter 7 presents a felling mechanism, attached to the machine, for tree felling. Contributions of this chapter are to present an autonomous tree felling mechanism. This mechanism can be mounted onto the mobility platform of Chapter 3 and deploys new sensing methodology, enhancing the felling process.

Chapter 8 concludes the thesis by summarizing outcomes and findings. Future work is also proposed.

Chapter 2

Literature Review

2.1 Robotics in the agricultural industry

A robotic arm system has been reported by Foglia and Reina [26] for the purpose of harvesting the vegetable radicchio. This system used computer vision from a monocular camera to localize the target. A linkage system was developed to cut and retrieve the vegetable from its growing position in the soil. The system was a static arm designed to be mounted to a moving tractor, with the ability to harvest one radicchio plant every seven seconds. Route planning was limited to the nearest neighbour, approached in the row selected by the operator. Due to a labor shortage of seasonal workers in Japan, a cherry-harvesting robot was developed by Tanigaki et al [27]. By combing the data from two pulsed-laser beams, the positions of cherries were calculated. Analysis of the data returned by the laser beam classified cherries and obstacles according to the components of the reflected light spectrum. Further agricultural developments include devices for cucumbers [28], apples [29] and strawberries [30].

From the literature previously cited, it has been shown that robotic harvesting is more than just a possibility; despite this, only limited developments have occurred in the forestry industry. Lam and Xu [31] describe a small, agile tree-climbing robot capable of maneuvering on a tree through the use of two grippers connected by a flexible body. The robot implements an autonomous climbing algorithm to explore the tree with goals such as animal observation or tree maintenance.

Devices for tree pruning have been an area of research interest, for example those in Fig. 2.1. A tree-pruning device developed by Soni et al [32] is positioned on a tree by a human operator. The device encircles the tree and uses wheels to propel itself along the

tree trunk. A five degree of freedom arm is mounted onto the platform that executes the pruning task. Ueki [33, 34] developed a pruning robot as “The timber industry in Japan has gone into decline because the price of timber is falling and forestry workers are aging rapidly.”. This machine encircles the trunk similarly to Soni et al. but utilizes it’s wheels to form hybrid motion between straight and spiral climbing. A chainsaw unit mounted on the machine was shown by experimentation to cut branches of up to 1 cm. Research interest in forest robotics has focused on pruning, measurement and exploration rather than on felling.

(a) “Treebot” [35] (b) Tree limb pruning robot [36]

Fig. 2.1 Two selected tree robots

Studies in humans show that "... movements become inaccurate and unstable when the "sense of touch" is lost." [37], by having this sense the machine will be able to guide its gripper more accurately around the tree leading to better contact and grasp. A review of tactile sensors has been carried out in [37] and depending on choice of sensor technology the gripper can gauge properties such as the size and shape of the tree, surface temperature and importantly, slip. Tactile sensing systems could include the use conductive rubber, capacitive based pressure sensors and resistive force sensors sampled at 25Hz to provide an indication on force [38]. Methods of estimating friction coefficients [39, 40] or classifying grip quality [41] can be used to avoid slippage or crushing when grasping.

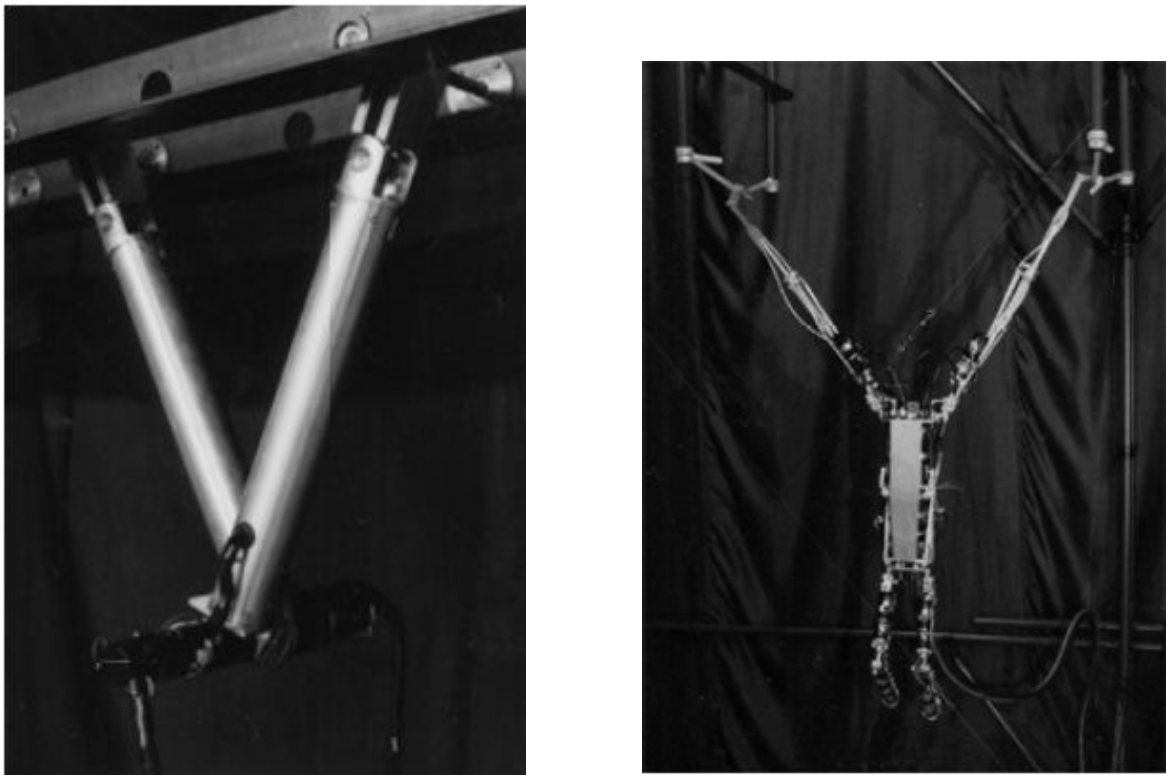
2.2 Brachiation in nature and machines

Brachiation is the process through which motion is achieved by swinging the body between points grasped with the hands. This motion is commonly associated with the Gibbon (Fig. 2.2). Laboratory studies such as Arms et al. [42] investigated the gait of arboreal locomotion styles in monkeys, particularly the grasping methodology through the use of X-ray. Through observation, their tendencies and capabilities in a range of environments was established which provides a base line for arboreal location performance. Usherwood and Bertram [43] analysed the gibbon's brachiation methodology by considering the energy throughout the process. It was shown that the loss of energy is correlated with the alignment of both the incoming and outgoing paths. The path is controlled however, by the gibbon moving the center of mass with leg flexion and extension. Their analysis identified the key factors of this locomotion such as arm bend, leg lift, exit angle and overshoot. Usherwood, Larson and Bertram [44] considered the dynamics of brachiation for a gibbon. They showed further that leg lifting and arm bending were significant factors in the process as well as the torque in the shoulder joint. A point-mass model of the locomotion has been derived by Bertram et al. [45] and pendulum based models are often used to describe the motion.

Fig. 2.2 Brachiation of a White-Handed Gibbon (Hylobates lar), adapted from [46]

Despite the research into brachiation in Gibbons, few machines have been developed to mimic this functionality. Fukuda and Saito [47] developed a two-link machine capable for swinging from hold to hold on a horizontally mounted ladder (Fig. 2.3). The control system implemented heuristic motion generation which controlled the swing amplitude. Meghdari et al. [48, 49] then expanded the control system of an equivalent two link machine to obtain trajectory planning for minimum control effort. A motion control system of a more complex machine was developed by Nishimura and Funaki [50]. Their machine featured three links,

weighed 5.74 kg and has a span of 0.798 m. A machine was also designed to mimic many more of the features of a gorilla [51]. This machine with 24 degrees of freedom making up the legs, torso and arms. Weighing 22 kg and standing 1 m tall, the machine was still capable of brachiating as well as walking and crawling.



(a) Brachiator 2 (b) Brachiator 3

Fig. 2.3 Two generations of brachiating machines [51]

2.3 Route planning for field robotics

Work to date on coverage path planning has considered applications involving machines such as tractors for field-based operations. The issue of spreading seed on fields using tractors is addressed by Jin and Tang [52] and Hameed [53, 54] covered other generic field maintenance tasks such as fertilizing. Hameed developed a planning approach that extends beyond flat fields to 3D terrain. This implements an energy-consumption model of the machine in order to plan the optimal driving direction. This approach minimizes the direct energy needed by the vehicle when it is traveling in parallel paths along the field. Jin and Tang [52] showed

how elevation models can be used to form analytical models by identifying subregions in the terrain. These subregions were individually optimized according to their unique features, as opposed to adopting a single solution for all terrain approaches. Once a path was found through the area, parallel work paths were formed by offsetting the curve to the edges of the workspace (Fig. 2.4). These planning approaches address the issue of covering a large, three-dimensional field with a tractor-type vehicle while seeking to maximize the area covered. The objective crops in previous schemes are in a highly structured environment consisting of rows or parallel paths that have simple geometries. The tasks of the machinery are generally 'area coverage' operations, for instance, that of spraying a field.

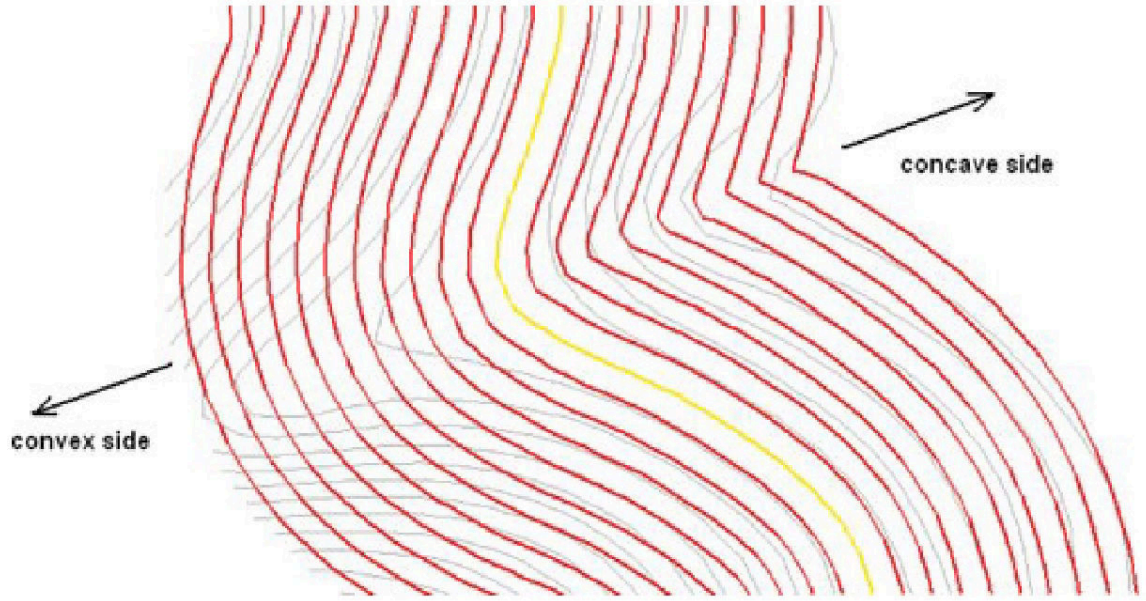


Fig. 2.4 Generation of parallel vehicle paths by offsetting a baseline path [52]

Bochtis et al [55] generalized the planning of orchard operations to the form of a graph problem. In their case, operations need to be done row-wise. Such tasks include mowing, spraying and harvesting. Indeed, Bochtis and Sørensen [56] show that vehicle routing-problems are a generalized case of the Travelling Salesman Problem. To form the graph, nodes were used to represent the start and end points of each row, and a corresponding matrix of connection costs was generated. Bochtis et al [57] also extended their approach to field planning for deriving a path for wildlife avoidance that was based upon escape models for the animals. For planning, Kang et al [58] used a machine-learning approach for stochastic node models, and Bao et al. [59] showed how machine learning can be coupled with computer vision and human interaction in planning paths, even in intricate environments.

The Depth First Search (DFS) and the Breadth First Search (BFS) by Gleich [60] are commonly used to traverse graphs. The BFS operates by first exploring nodes on the same level before investigating their children, whilst the DFS will always explore children before returning to the level above. The differences between the DFS and the BFS come down to the data structure used to store the discoveries: in the DFS it is a stack, whereas the BFS uses a queue [61]. The Travelling Salesman Problem (TSP-NN), following Kirk's [62] implementation, determines the route by always visiting the next closest node until the end node is reached. Fatime, [63], presented a graph-searching algorithm to find the specified length between two nodes. Fatime's approach, however, suffered from having to maintain arrays that grew too large for a computer to store.

In order to manage large commercial forests, operators require many metrics on its performance: using both "traditional data types and remotely sensed data is widely recognized as essential and critical for extensive 'on-time' environmental studies and monitoring at the local, regional (landscape), and national levels" [64]. Due to the scale of these operations, accessibility can be an issue; hence, it is common for data to be collected aerially for sources such as satellites, aeroplanes, helicopters and balloons. LiDAR and images at a variety of wavelengths are commonly returned data. Maltamo et al. [65] described many of these applications for airborne LiDAR. Analysis of this data is used to provide the end user with information of "forest area, volume, condition, growth, mortality, removals, trends, and forest health" [66]. LiDAR has been proven effective in estimating the biomass, canopy volume and mean height of large footprint systems by Lim et al. [67], whilst small footprint systems are able to inform the user on the crown width and height of individual trees. Culvenor's work [68] shows how high-resolution imagery can be used to delineate tree crowns (Fig. 2.5). By considering the changes in brightness between the center and edge of the crown, in the near infrared, red and green bands, container cells are formed. These cells contain the detected canopy and the tree crown's location can be found using a local maxima search.

2.4 Environment detection

Previous work on tree identification using LiDAR has resulted in models of trees of limited applicability to mobile robot platforms - particularly for navigation and commercial task execution. Work has investigated deriving complex models of individual trees using multiple scans from different locations, see Figs. 2.6 and 2.7. Firstly, registration must be performed between subsequent scans in order to form one coherent cloud [69, 70]. Calculation of the

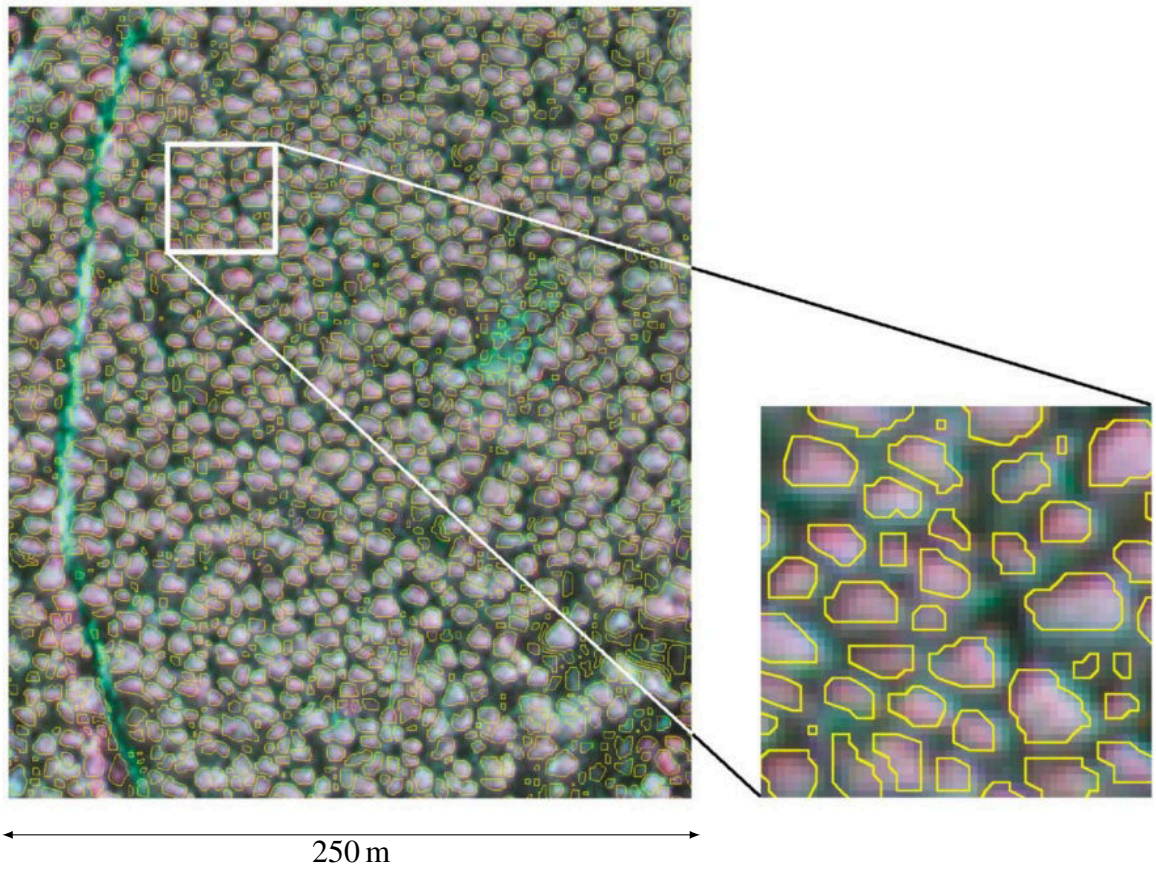


Fig. 2.5 Tree delineation from airborne video, adapted from [68]

biomass and generation of accurate 3-D models of trees has been achieved by [71, 72, 73]. However, these derive overly complex models not required for navigation or are unsuitable due to their requirement for scans from multiple viewpoints to generate the model. Work by McDaniel et al [74] used a LiDAR unit on a manually adjustable base to sample the environment and recognize trees. This approach required detection of the ground surface first, then modeled trees using two horizontal slices through the point cloud at a height relative to the estimated ground surface. In a practical harvesting scenario the ground surface is often significantly obscured which can result in a poor estimation of the surface. In the cited work, their scans were performed on terrain much more regular than can be expected of a commercial environment with obstacles such as ground cover, wind/dead fall, unpruned trunks and dense canopy.

Other work has investigated the modelling of forest environments using aerial based LiDAR [66, 67, 75]. Pont et al. [76] present a tree counting methodology using such aerial LiDAR. The system was evaluated with stand of 5 to 32 years of age, although limited to

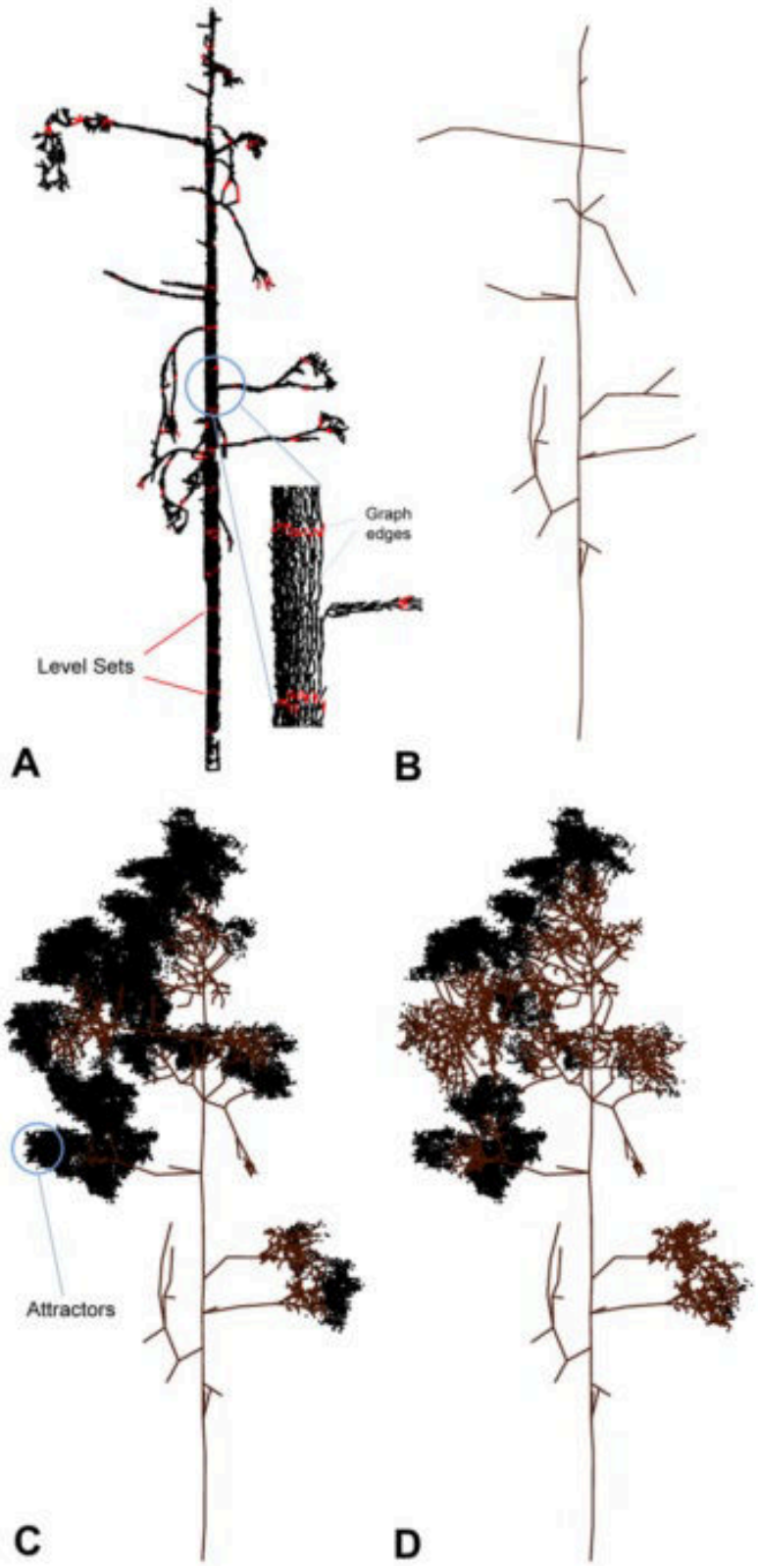


Fig. 2.6 The structural model generation process as in [71]

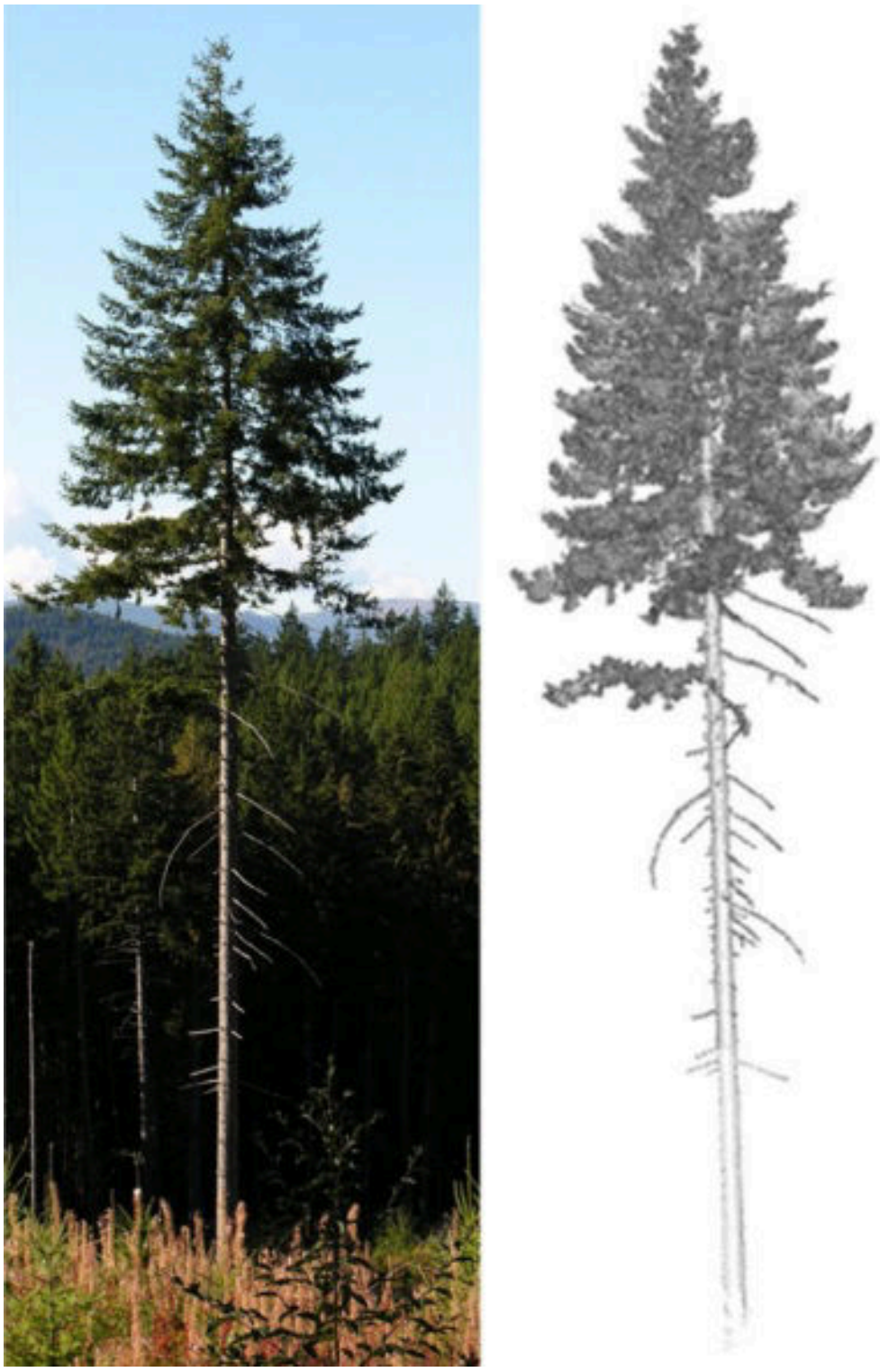


Fig. 2.7 Photograph and registered point cloud for a Douglar fir tree, adapted from [71]

stands that have a homogeneous tree age. An error of 4.7 % to 6.1 % was achieved. Wallace et al [77] developed a novel LiDAR instrument and shows that leaf and bark areas may be measured. Wallace, Lucieer and Watson [78] developed a platform using high resolution LiDAR and an Unmanned Aerial Vehicle. This platform was used to detect and delineate trees, comparing five different algorithms. It was found that up to 98 % of the trees were detected aerially and that a high coupling exists between the rate of omission and point density.

Trees have been used to provide coarse map localization previously. Hussein et al developed a system to coarsely define the position of a rover in cases of GPS outages [79]. Starting with an aerial image of the region of interest, tree stem locations were estimated from canopy apexes. As the rover moved through the environment, 3-D LiDAR data is used to detect trees. An Iterative Closest Point matching algorithm estimates the location of the rover. Their tests achieved an average accuracy of between 4.76 to 6.83 m. The primary source of error was the aerial imagery which had a 2 m RMS error in the horizontal plane. Ali et al [80] develop a visual method of tree detection for an autonomous ground vehicle (Fig. 2.8). The system identifies tree stems and estimates the distance to the vehicle, to be used for obstacle avoidance. Mounted on an Unmanned Aerial Vehicle (UAV), LiDAR has been shown effective for obstacle avoidance within a forest by Jin Qiang et al. [81] and over a forest with landmark tracking by Pinage et al [82].

2.5 Cutting devices for robotic harvesting

Studies such as that by Visser and Spinelli [83] help to determine the optimum stage at which felling should be performed. Their work showed that productivity varies significantly with log volume and for Pine the optimum piece size to be between 1.9 m^3 and 2.5 m^3. On an age basis, the economically optimum felling time occurs between 25 to 30 years of age according to the New Zealand Ministry of Forestry [16].

Digital sensor integration on chainsaws has been generally limited to detection of kick-back, or vibrations transmitted to the user. Arnold and Parmigiani [84] integrated a gyroscope to detect kick-backs within 30 ms. Osborne, Arnold and Hall patented the gyroscopic system [85]. Long term exposure and use of a chainsaw has been investigated by Peterson, Brammer and Cherniack [86]. Their work used an accelerometer and force sensor to measure the

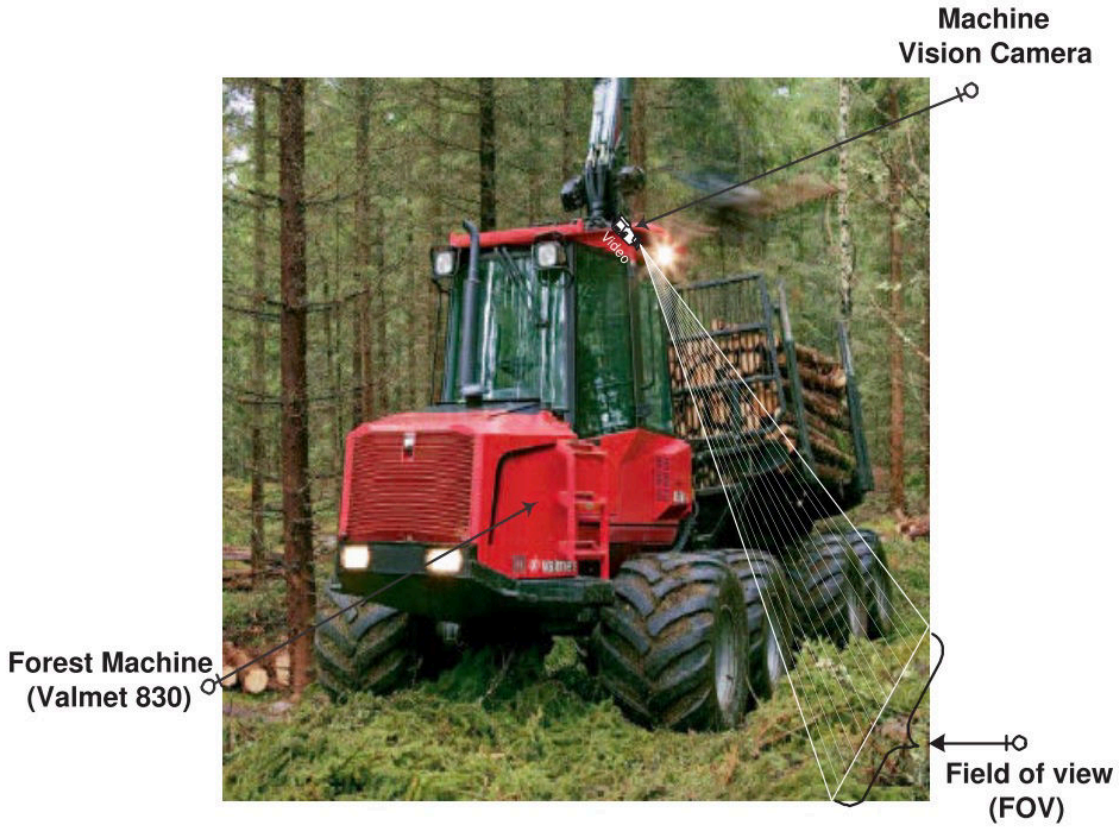


Fig. 2.8 Configuration of detection system on-board a Valmet 830 Forwarder [80]

transmission of vibrations from the device to the user over 8 hour workdays.

The use of chainsaws on robotic platforms has had little investigation. Ishigure, Kawasaki, et al [36, 34] present a robot that can be used to prune tree branches. Their lightweight platform can not fell trees, but rather clamps around them and uses a small chainsaw to cut branches with diameters less than 0.05 m. Their chainsaw uses a built-in wedge to prevent bite. Monitoring the current provides the system with an indication of the cutting progression; a threshold for the raising and falling edge of current allows the system to identify the start and end of the cutting procedure. Literature from Li et al. [87] shows that cutting discs inspired from biology can have efficient impact on the cutting process. Their work resulted in the design of a circular saw with teeth inspired from a mole rat, this design reduced normal stress by 23%.

Chapter 3

Brachiating mobility platform design

In this chapter, the robot is introduced. Firstly, the requirements for the machine's design are identified. The mechanical design is then elaborated, to demonstrate the design and construction of the novel machine. The locomotion methodology used by the machine to traverse the forest is described finally. The contributions of this chapter are to introduce the novel design of a forestry machine, as well as the new locomotion methodology not previously investigated.

3.1 Introduction

Due to the dangers associated with felling trees in rugged environments Scion, in partnership with the University of Canterbury, are developing a teleoperated, semi-autonomous tree felling machine [88]. The machine prototype is designed to operate in commercial forestry environments, such as that shown in Fig. 3.1. The role of the machine within these environments will be to fell trees, thereby moving the forestry worker from the dangers of felling a tree personally, to a supervisory position controlling the machine.

3.2 Requirement specification

The requirements for the machine must first be identified before design can occur.

The foremost requirement is the scale of the machine. Due to the impracticalities of building a full-scale felling machine, one should be constructed to a size suitably small for laboratory testing. However, the machine should also be suitably large such that field testing can occur in commercial forests. This requirement determines characteristics of the system

Fig. 3.1 Battle Lake Pine forest, Christchurch, New Zealand

such as length, width and height but also the target tree size.

A leading problem with existing ground based machines is the damage they cause to the terrain, as identified in Chapter 1. Given also that much forest is located on steep terrain [6], it is required that the machine must travel through the forest independently of the ground.

To address the safety concerns seen in both motor-manual and mechanized harvesting, the machine must be teleoperated. Therefore communications to facilitate control as well as suitable sensing technology must provide the operator with adequate information to do so.

In order to harvest trees, the machine must carry a felling mechanism. This mechanism is required to cut trees with diameters based upon the scale of the machine.

The requirements can therefore be summarised as such:

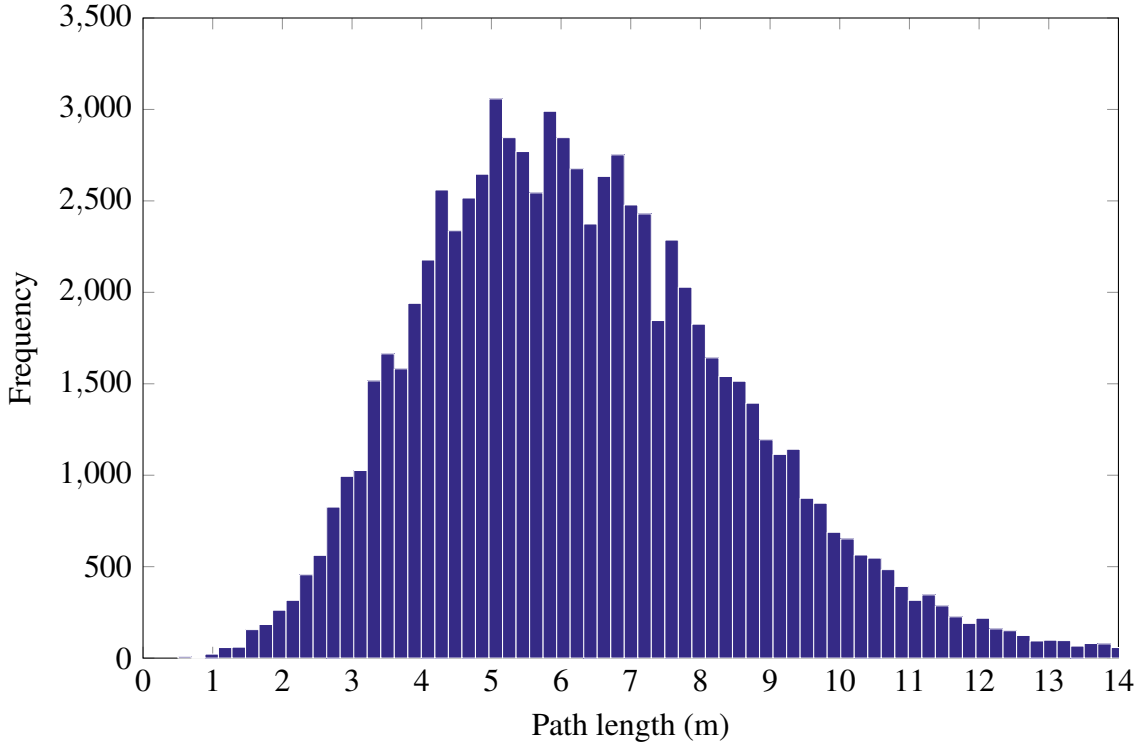
- Laboratory scale

- Ground independent
- Teleoperated
- Felling capability

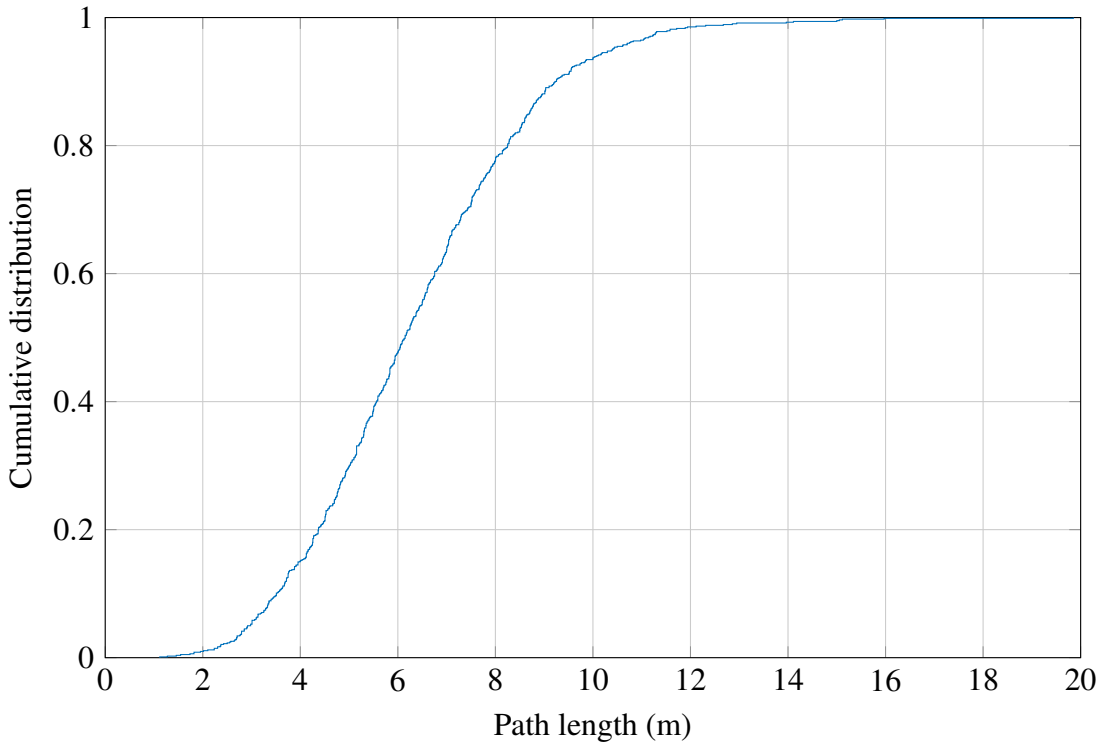
The machine's scale is the principle requirement to determine. In the case of ground independence (using brachiation), the reach of the machine must also be sufficiently long to bridge gaps between trunks. To be effective in the commercial environment, it should be capable for visiting at least 80 % of trees. Any trees that the machine could not operate on can be serviced manually by ground crew motor-manually.

In order to specify the required reach of the machine, an understanding of the distances between trees must be gained. A statistical investigation was performed on a representative plantation forest. This forest [89] consisted of 13 829, 27 year old Pine trees, ready for harvesting in the New Zealand area. Each tree was tagged using GPS to obtain it's position. Tree to tree paths were found by applying the Delaunay triangulation [90]. This triangulation methodology is selected as it "... connects points in a nearest-neighbor manner." [90]. A histogram of the path lengths is shown in Fig. 3.2a. Results show that the mean and median path lengths were 6.42 m and 6.15 m respectively. The standard deviation is 2.41 m. From the cumulative distribution function, plotted in Fig. 3.2b, it can be seen that 80 % of paths are less than 8.3 m. For this length to be practical for use in the laboratory, the scale of one-quarter is selected. This specifies the length of the machine to be 2.075 m.

In order to grasp trees, a gripper must have a sufficient clearance to allow the trunk to enter within its grasp. Similarly to the reach requirement above, the grasp is analyzed. A dataset consisting of 444 trees [89] was analysed. This dataset contained information of a forest stand with trunk diameter measured at double breast height, as well as GPS coordinates for each. A histogram of these diameters is shown in Fig. 3.3a. The mean and median diameters were 388.73 mm and 388 mm respectively. The standard deviation is 66.46 mm. From the cumulative distribution (Fig. 3.3b), the grasp to meet the requirements for 80 % of trees is 445 mm. Therefore, by applying the scale identified above ($\frac{1}{4}$), tree diameters are expected to be 111.25 mm. This represents the minimum grippers size, as well as the diameter the felling mechanism must be capable of cutting.

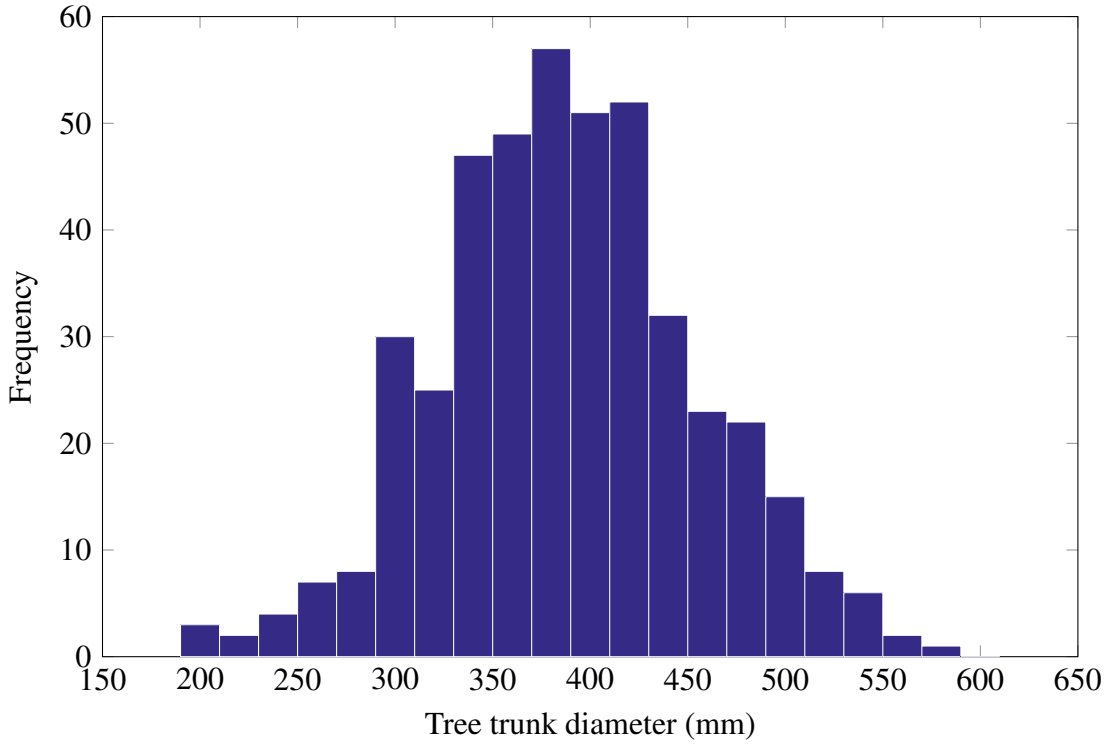


(a) Path lengths between trees

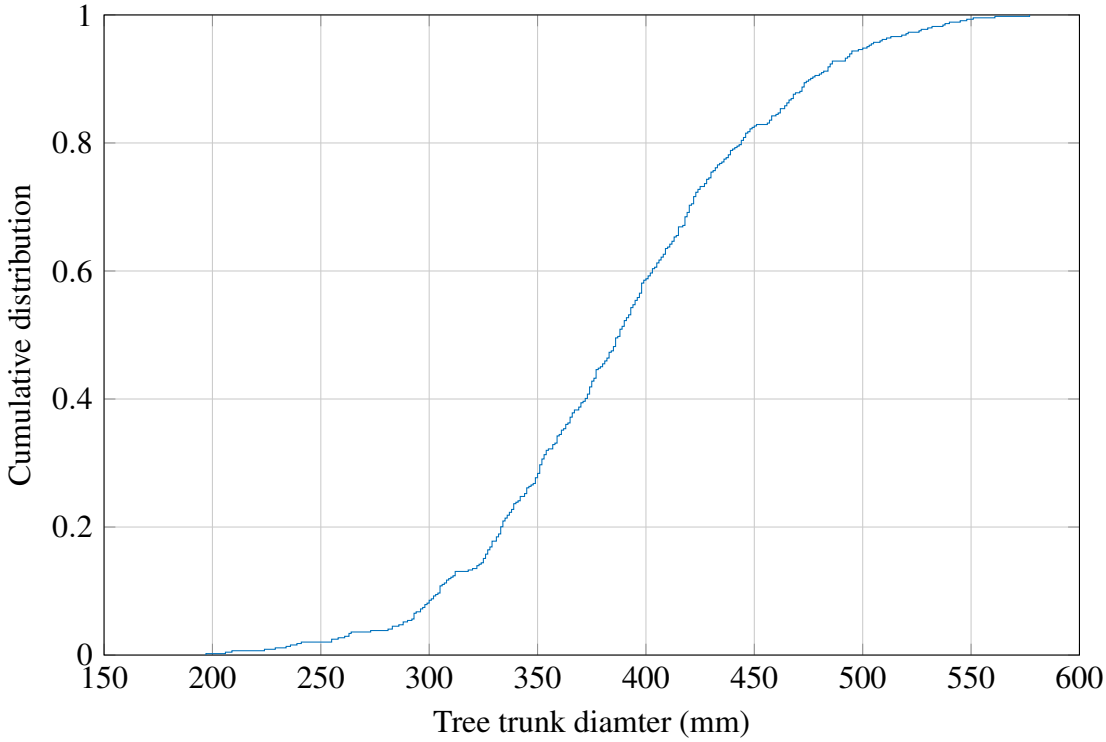


(b) Cumulative distribution of path lengths

Fig. 3.2 Inter-tree path lengths in forest



(a) Histogram of tree trunk diameters



(b) Cumulative distribution of tree trunk diameters

Fig. 3.3 Tree trunk diameters

3.3 Machine construction

For this machine to lead to an understanding of the practicalities of a full-scale system, a $\frac{1}{4}$ scale prototype is developed [91]. Justification for this scale was given in Section 3.2. This scale allows for practical on-site laboratory testing at the University of Canterbury as well as field tests in young, un-thinned forests. An image of the proposed design is shown in Fig. 3.4. To meet the low weight requirements and budget constraints, aluminium was used as the primary building material.

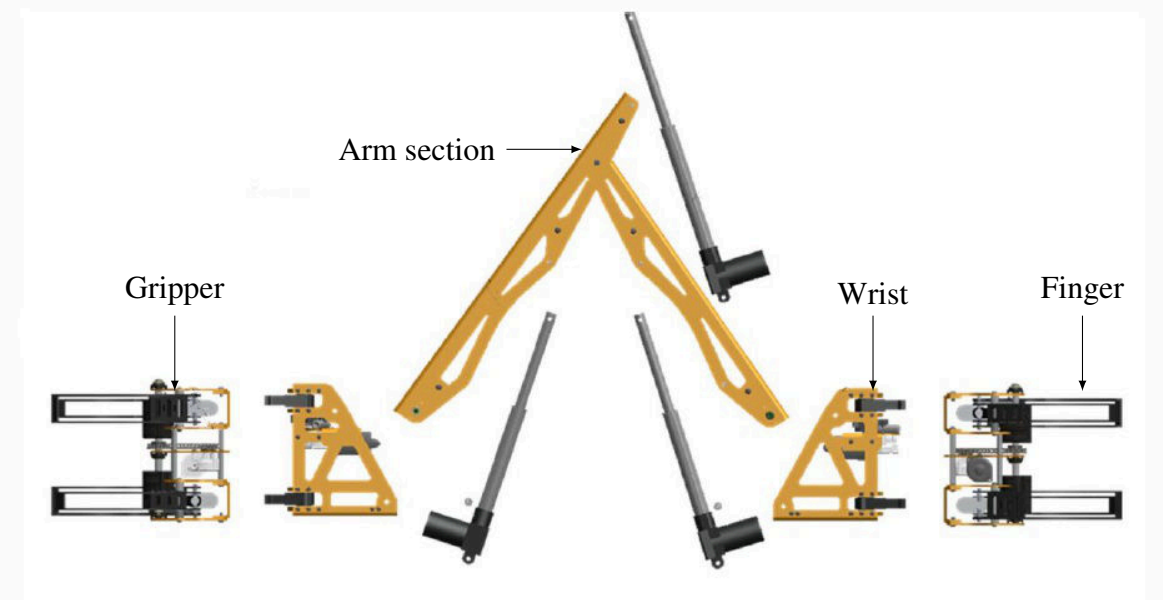


Fig. 3.4 Exploded view of tree-to-tree machine, adapted from [91]

The design consists of three basic modules; grippers, wrists and arms. Overall 11 degrees of freedom are implemented; three in each gripper, one in each wrist and three in the arms. From it's base gripper tool centre point (TCP), the TCP of the opposite gripper can be

Table 3.1 Machine properties

Property	Value
Displacement (Horizontal)	1.0 m to 2.2 m
Displacement (Vertical)	−0.75 m to 0.75 m
Graspable tree diameter	50 mm to 435 mm
Maximum felling diameter	220 mm
Weight	75 kg

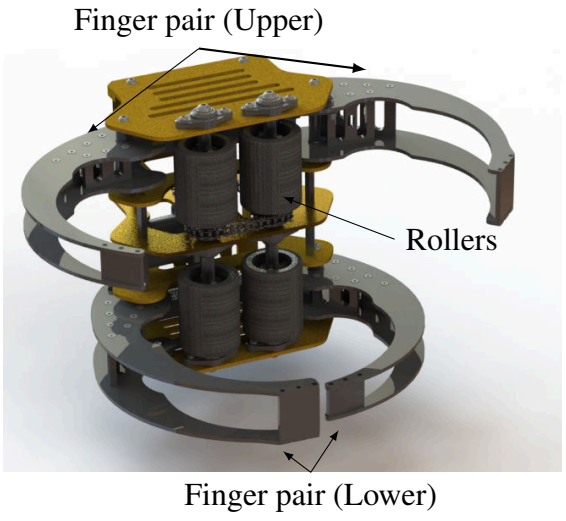
displaced from 1.0 m to 2.2 m horizontally and -0.75 m to 0.75 m vertically.

Grippers (Fig. 3.5) are located at either end of the machine. This allows grasping of up to two trees at once. Each gripper contains two pairs of fingers located at the top and bottom. Each pair is independently actuated in order to allow grasping of objects, primarily tree trunks, which may have irregular surfaces. The fingers are designed to wrap around the back of objects, pulling them into the grasp. For the front of the tree trunk interface with the gripper is a set of rollers. These rollers have a rubber coating which provide a semi-deformable surface, allowing a secure grasp whilst limiting damage to the tree bark. The function of the rollers is to provide a mechanism by which the machine can rotate about the grasped tree (Fig. 3.6). These rollers would allow the machine to rotate 360° for the base tree in order to align to the next tree.

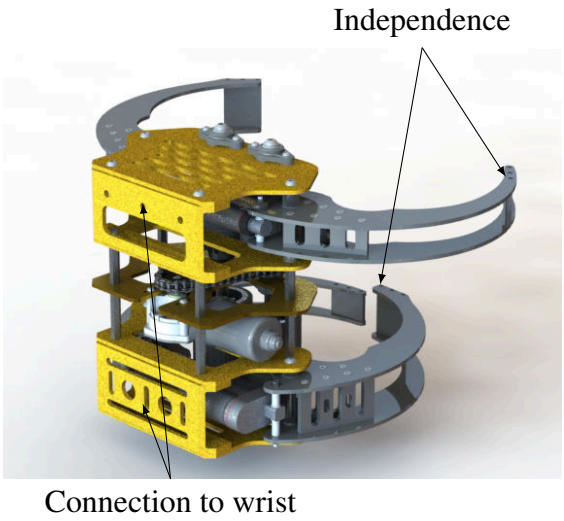
One wrist (Fig. 3.7a) is attached to each of the gripper modules in order to connect them to either end of the arm module (Fig. 3.4). Each wrist contains a revolute degree of freedom to rotate to gripper relative to the wrist. Rotation is achieved through a gearing system (Fig. 3.7b) with a ratio of 2.5:1. Motion of the wrist motor can have one of two effects; if the gripper attached to the wrist is free (not grasping) then the gripper rotates, whereas if the gripper is not free then the robot rotates as a result.

The arm module, shown in Fig. 3.4, connects to both of the wrists to complete the kinematic chain between both grippers. Detailed views are shown in Fig. 3.9. In this module, three linear actuators are contained to manipulate the gripper position. These actuators move the free gripper, “end effector”, in the two dimensional plane coincident with all three linear actuators. Three dimensional positioning is achieved using the rotary joints of each wrist.

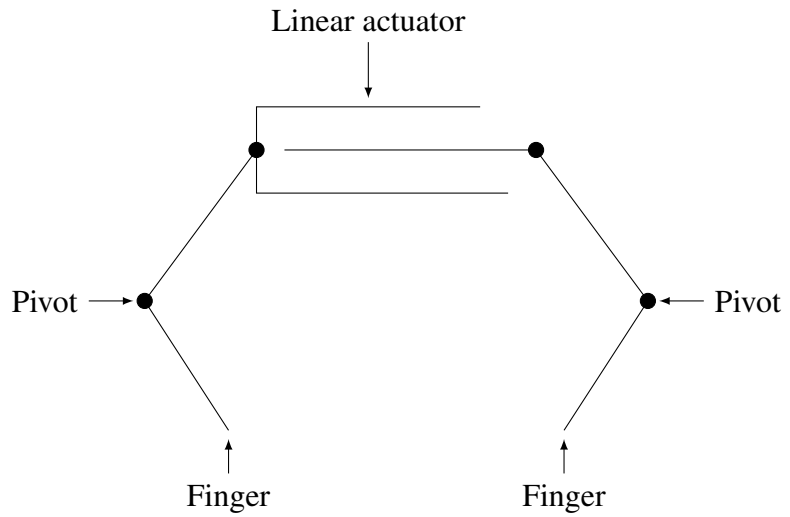
The machine operates as a quasi-static cantilever as shown in Fig. 3.8b. The weight of the machine generates a turning moment M_o given by the distance of the center of mass d_{CoM} and the force F_w. The gripper is responsible for maintaining the mechanical stability of the machine to the trunk. Therefore each gripper must be capable of providing the reaction moment M_r to counter M_o. This is provided by the finger strength, F_r and the moment tuned by engineering the distance between fingers d_f.



(a) Front view



(b) Rear view



(c) Kinematic diagram of gripper from top view

Fig. 3.5 Gripper design

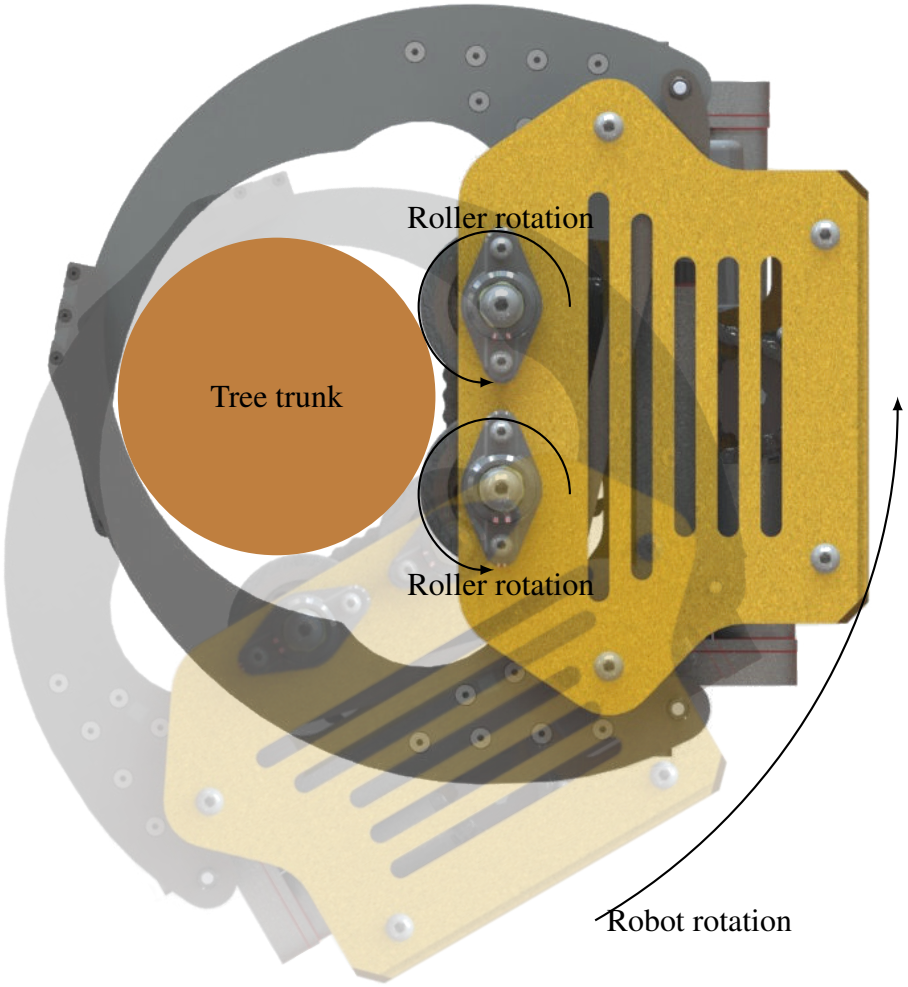
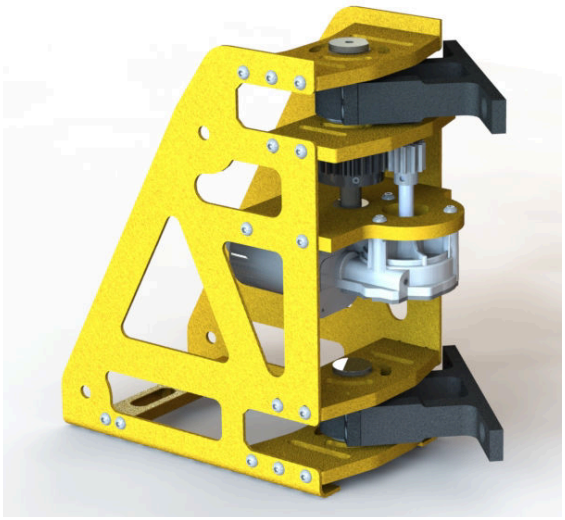
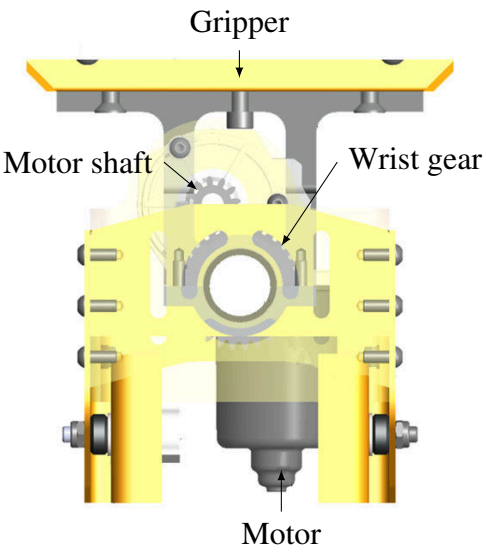


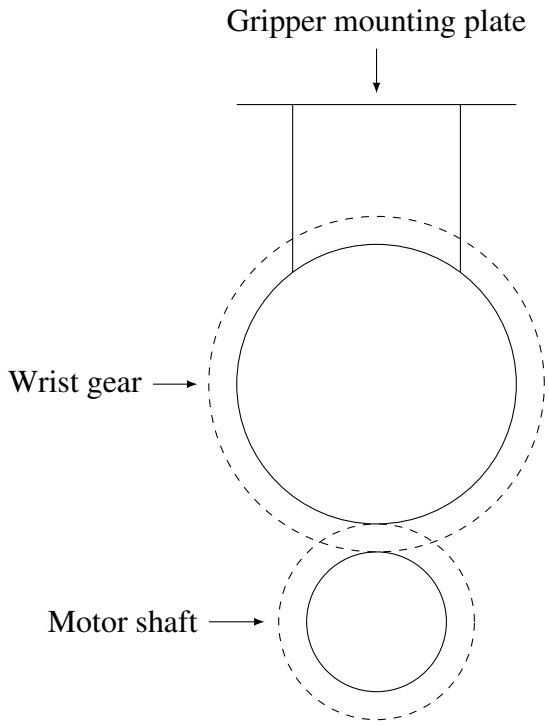
Fig. 3.6 Method of rollers to rotate about tree trunk



(a) Perspective view

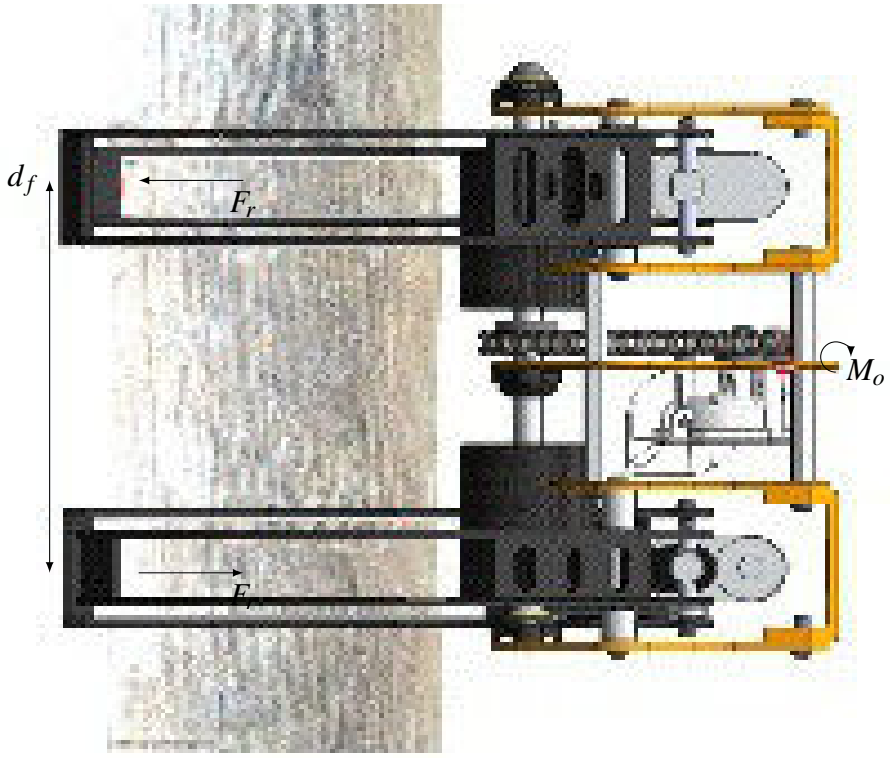


(b) Top view showing Wrist-Gripper connection, adapted from [91]

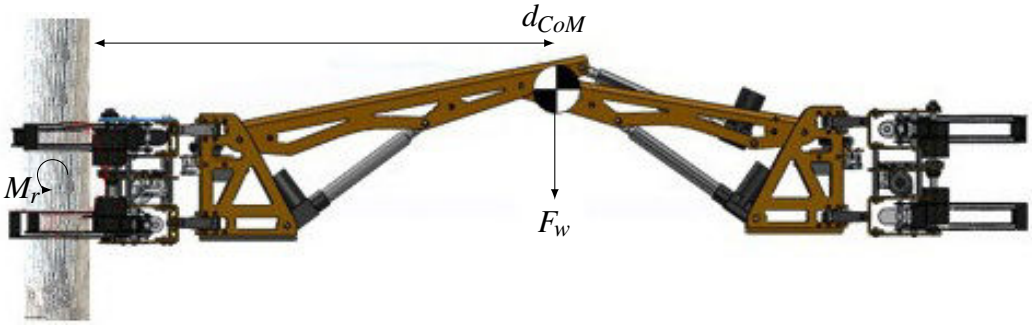


(c) Kinematic diagram of wrist from top view

Fig. 3.7 Wrist design

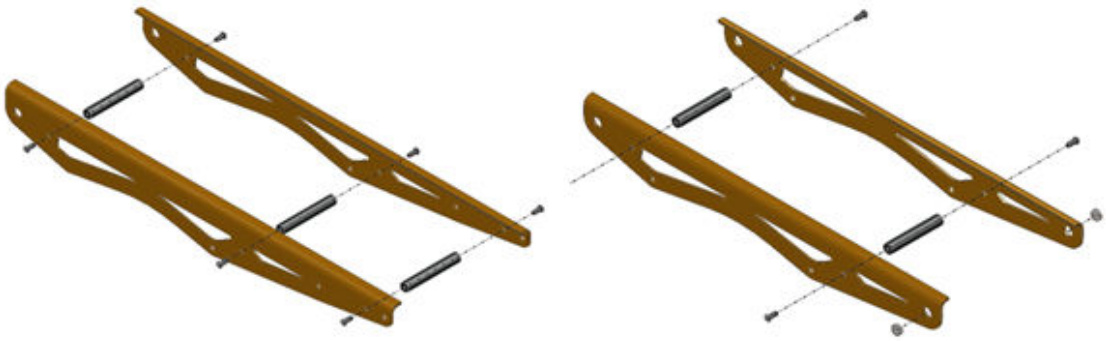


(a) Gripper

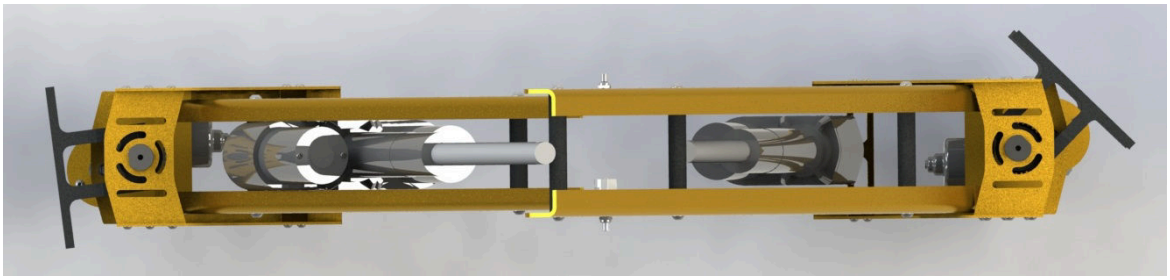


(b) Arm

Fig. 3.8 Forces and moments in the machine, adapted from [91]



(a) Exploded assembly [91]



(b) Top view showing arm and Wrist-Arm connections

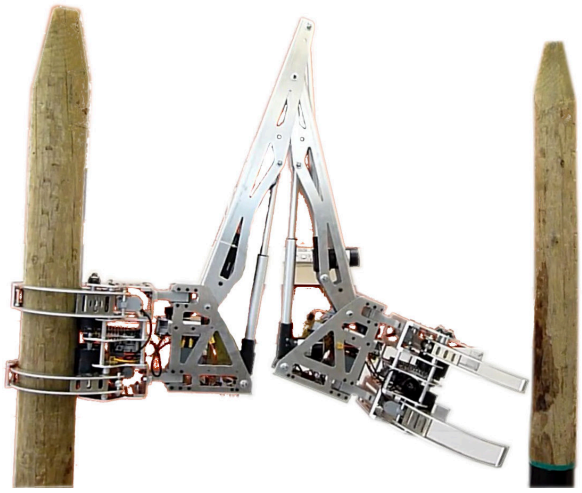
(c) Front view of arm and Wrists

Fig. 3.9 Arm design

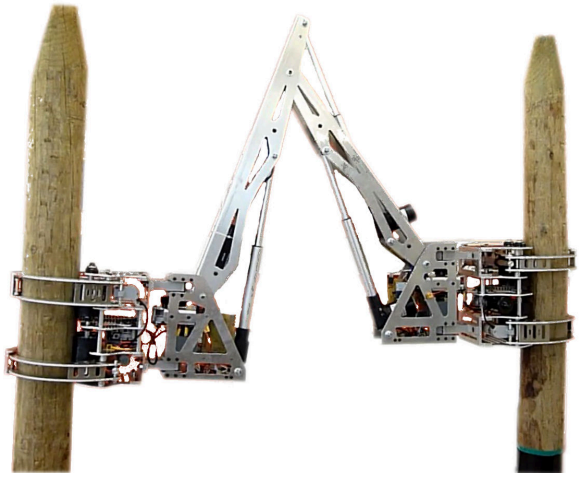
3.4 Brachiating locomotion methodology

A method of brachiation, arboreal location seen in primates (Fig. 2.2), is selected for this application. This achieves the requirement (Section 3.2) of ground independence. Work by Papesch et al. [92] verifies the ability for Pine trees to mechanically support such a machine for brachiation. During locomotion, the machine requires at least one gripper attached to a tree at any time in order to support against gravity. The manoeuvrability of the proposed machine results in a more readily deployed machine over alternatives such as the ClimbMAX (Fig. 1.7). Travel is not limited by a cable, as is the case with the ClimbMAX.

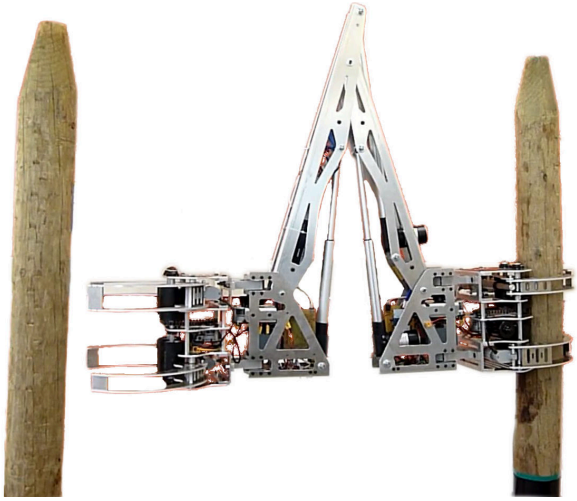
The steps taken by the machine to traverse between two trees is shown in Fig. 3.10. The machine is first positioned into the forest with a vehicle. With the vehicle adjacent to the starting tree, the operator can manually manipulate the machine in order to obtain a grasp on the tree. From this position it can now be operated throughout the forest. Locomotion begins with this initial gripper fixed to the tree, the opposite gripper (end-effector) is then free to move (Fig. 3.10a). Through operation of the wrist's revolute joints and the arm's linear actuators, the robot is rotated and extended to manipulate the end-effector position. Turning the end-effector toward the next tree using the wrist motors aligns the gripper with the target. The arm module can then be extended towards the tree through a coordinated movement of the linear actuators. When close to the desired tree, the revolute joint of the free wrist is used for fine, local alignment of the gripper to the tree. Upon reaching the next tree, fingers close around the trunk to secure its grip. At this interim stage, the robot is support at either end (Fig. 3.10b). Finally the initially static gripper is released, therein completing the transfer of the robot from one tree to another (Fig. 3.10c). By repeated cycles of grab and release, the machine traverses the forest by visiting one tree after another. The machine can been seen performing this task in a commercial forest in Fig. 3.11, during it's interim motion stage.



(a) Starting position



(b) Intermediate position attached to both trees



(c) Release of initial tree, completing transfer

Fig. 3.10 Traversal of the robot, adapted from [88]

Fig. 3.11 Tree-to-tree machine in a Pine forest

3.5 Felling capability

A chainsaw mechanism mounted below the gripper (Fig. 3.12) facilitates tree felling. The felling process is started whilst the machine is grasping trees at both ends in order to provide stability (Fig. 3.10b). The underslung chainsaw advances into the trunk according to a felling policy based upon sensory data and user input regarding the tree. These policies and data requirements are detailed in Chapter 7. When the tree is detected to be on the verge of falling, the gripper releases the tree (Fig. 3.10c). This leaves the robot supported only from the next tree and unattached from the tree being felled to allow it to fall. Cutting continues, however at this stage the robot is now detached from the tree which allows it to fall freely. The cutting policy designs the cuts such that the tree falls away from the machine, this is based upon the lean of the tree.

3.6 Control system

A distributed control system is used to operate the robot. The system architecture is a master-slave configuration involving a computer on-board the robot and a base station for the operation, communicating over a private, ad-hoc Wi-Fi network. The system master controller is that of the robot, allowing the robot to reach a safe state if there is an outage

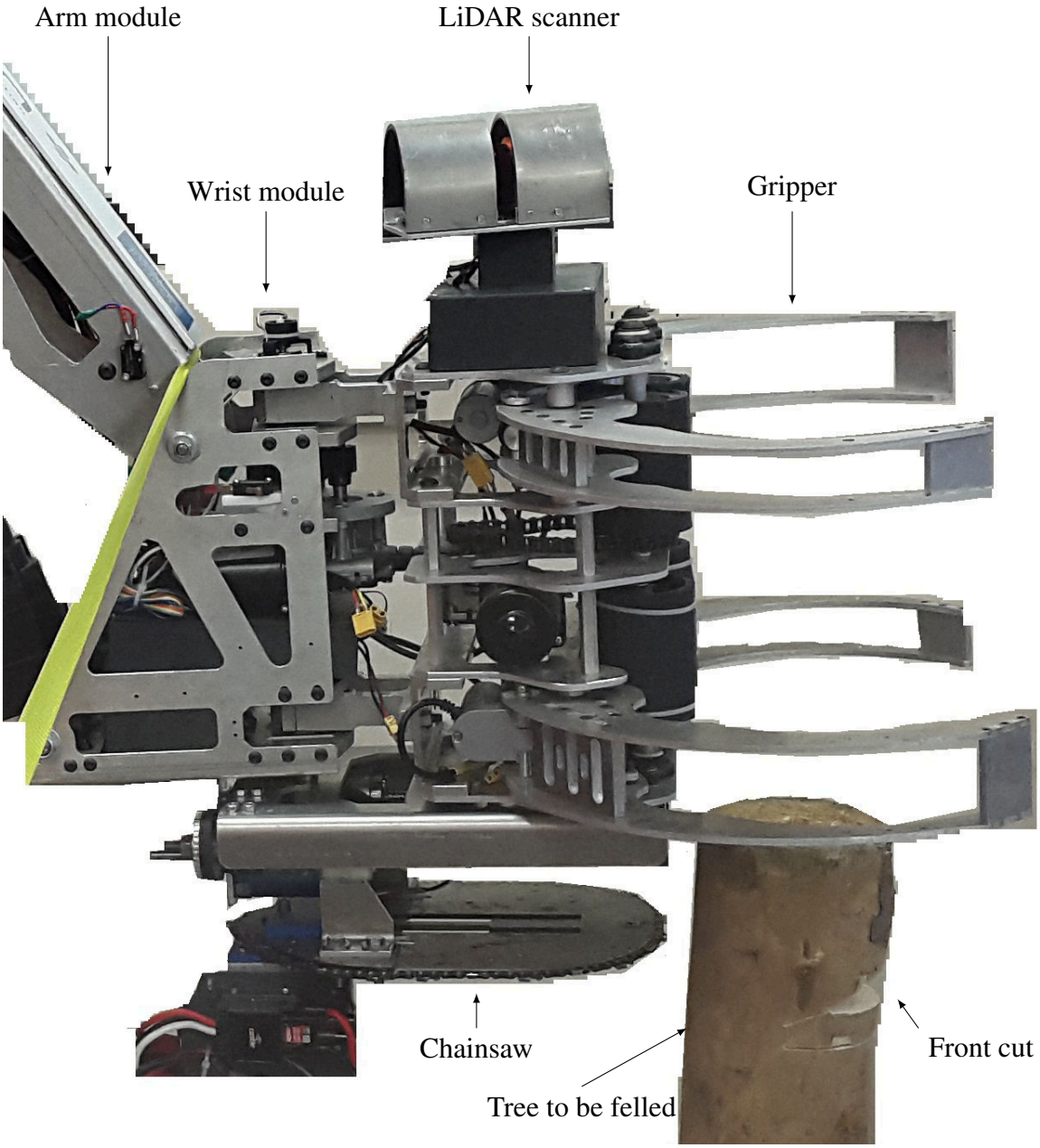


Fig. 3.12 Chainsaw, gripper and wrist assembly

on the network. Therefore the operator terminal runs in a slave configuration, requiring the robot to offer the server connection.

The software package Robot Operating System (ROS) [93] is used throughout the control system. “The Robot Operating System (ROS) is a flexible framework for writing robot software. It is a collection of tools, libraries, and conventions...” [93]. In using ROS, an extensible framework is available allowing software libraries to be developed which incorporate the systems, such as message handling, for distributed computer systems. A flow chart showing data flow and node connectivity in the control system in shown in Fig. 3.13.

An operator controls the machine from a remote base station. A display (Fig. 3.14) indicates the current store of the robot and detected objects from the environment. The operator can control the actuators of the machine using one of two mechanisms; manual joystick or semi-autonomous motion planning. In using the joystick, an operator is able to manipulate actuators individually. Using software however, the operator can specify the desired end-point for the robot and a motion path can be automatically generated and executed. In the latter case, the operator is only needed to supervise the machine, whilst the robot performs the motion plan automatically. Motion plans are generated using the Open Motion Planning Library (OMPL) [94], this allows environmental knowledge from the LiDAR scanner to be used to avoid collisions.

In order to ensure fast, robust and responsive control of the robot, computationally heavy tasks are performed by the base station. Therefore a high complexity controller is not required on-board, reducing costs and power consumption. Data required for processes such as object detection from point clouds or route planning is transmitted to the base station using the Wi-Fi network.

As the network is based upon Wi-Fi, this allows other machines to operate in the same space without hindrance and allows other operator users to access one or many other machines in order to have a single operator responsible for more than one machine.

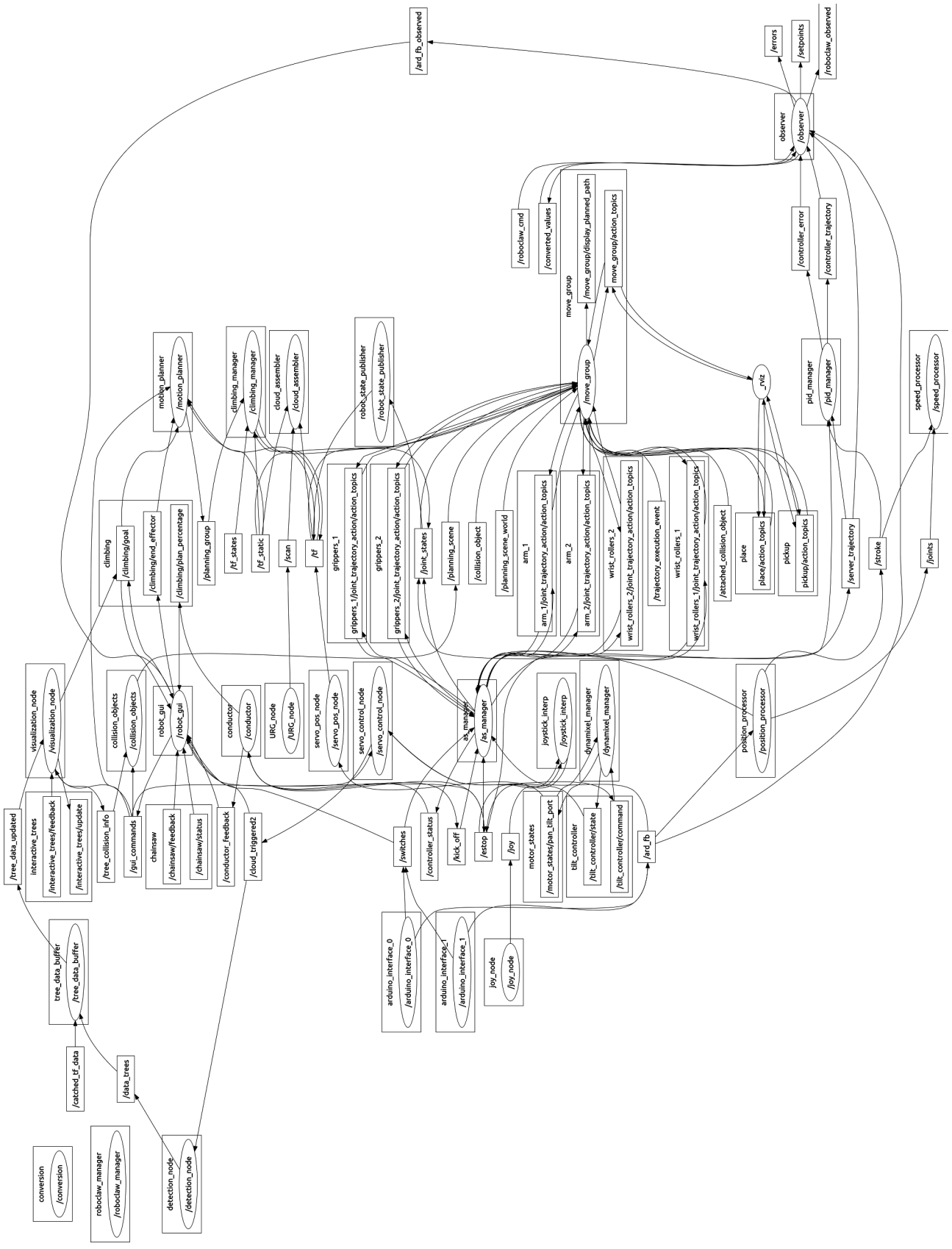


Fig. 3.13 Overview of ROS control system

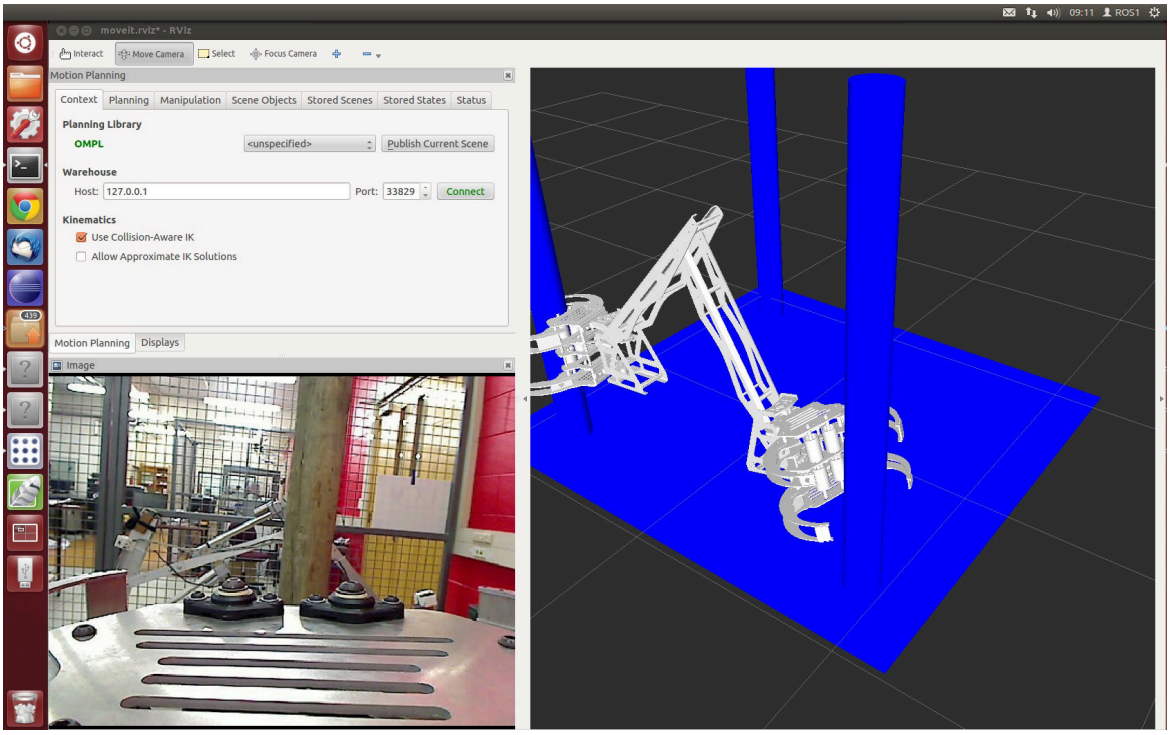


Fig. 3.14 Typical view in remote operators terminal. View includes video feedback (bottom left), path planning functionality (top left) and current state visualization (right).

Chapter 4

Quasi-static analysis of a bipedal brachiating machine

A novel brachiating, biped mobility platform for tree to tree traversal is analyzed by its kinematics and dynamics. The machine performs brachiation in the horizontal plane. Firstly, the kinematic model is derived then used to investigate the workspace of the end effector. The optimal grasping location is identified through the Yoshikawa manipulability measure. Secondly, actuator performance and limitations are identified through dynamic analysis of joint torques. Pose optimization is identified and the coupling with inertia is determined.

4.1 Introduction

Despite the biological studies on brachiation, an understanding of the mechanics of this locomotion for machines is underdeveloped. The machine implements brachiation in a new plane, not previously developed and weights three times more than previous machines (63 kg) such as the Gorilla in [51]. Operation in the horizontal plane, normal to gravity, has to the author's knowledge, not yet been considered. By traversing in this plane, the machine can work in a plantation forest and is capable of doing useful work such as felling, measuring and mapping of the forest inventory that was not possible when brachiating in the perpendicular plane.

In order to quantify the ability of a robotic mechanism to arbitrary change position and orientation from a given pose, Yoshikawa introduced a Manipulability Index [95]. This index can be used to find the optimum postures for manipulators. Analyzing this metric can benefit robot design and control because it can identify areas in which mobility is compromised such as at, or around singularities. Yoshikawa bases the index (w) upon the velocity manipulability

ellipsoid (Eq. (4.1)) which is formed by the velocity (v) and Jacobian at the current pose ($J(\mathbf{q})$). This ellipsoid indicates the "attitude of a manipulator to arbitrarily change end-effector position and orientation" [96]. The volume of the ellipsoid (Eq. (4.1)) is proportional to (Eq. (4.2)) which defines the Yoshikawa Manipulability Index [95]. Since the index, by the use of (Eq. (4.2)), reduces to zero at singularities it can also be used as an indication of the distance from such singularities and therefore by maximizing this value along a trajectory the redundancy of the manipulator can be fully exploited.

$$v^T \cdot (J(\mathbf{q}) \cdot J^T(\mathbf{q}))^{-1} \leq 1 \tag{4.1}$$

$$w = \sqrt{|J(\mathbf{q}) \cdot J^T(\mathbf{q})|} \tag{4.2}$$

4.2 Machine kinematics

In this section the kinematic model of the system is formed and then used to derive the workspace of the end effector. The transformation between base and end effector is shown in Section 8.2. The Yoshikawa manipulability [95] of the end effector is calculated over the workspace; the subsequent analysis identifies the optimal grasping location.

The brachiating machine proposed in Chapter 3 contains eleven degrees of freedom and is built from three primary types of rigid bodies (Fig. 3.4); two 'grippers' at the extremities, two 'wrists' connecting these grippers to the midsection and finally the 'arms' which connect both wrists. The grippers contain two fingers located at the top and the bottom whose positions are mutually independent as they are individually actuated which allows the gripper to close around irregularly shaped objects. The wrist component connects the gripper to the arm and contains a degree of freedom such that the machine can rotate about the gripper. Finally, the arm assembly contains two linkages and three joints generating the reach of the machine.

The machine achieves locomotion by brachiation (Fig. 3.10) which allows it to travel independent of ground conditions, causing no erosion to the soil. The traversal methodology begins with one gripper fixed to a supporting object (e.g. tree 1). With one gripper fixed, the free gripper is manipulated to such a point that it can close about the next supporting object (tree 2). Upon reaching the target object, the free gripper closes two pairs of fingers around the object (tree 2). At this point the machine is supported by both grippers (attached to tree 1& 2). The maneuver between the two objects is completed when the gripper is freed from

the original object and the machine is fully supported by the target object (tree 2).

For the purposes of forming an analytical model of the machine the origin, joint and links are defined as in Figs. 4.1 and 4.2. To simplify the inspection of the machine position by humans, the origin and end effector coordinate frames are located at the base of each gripper, centered along its grasping face (Fig. 4.1). As the linear actuators are used to produce a rotation in the joint, joints 3,4 and 5 are modelled as revolute rather than prismatic joints. This simplification also removes the closed kinematic loops.

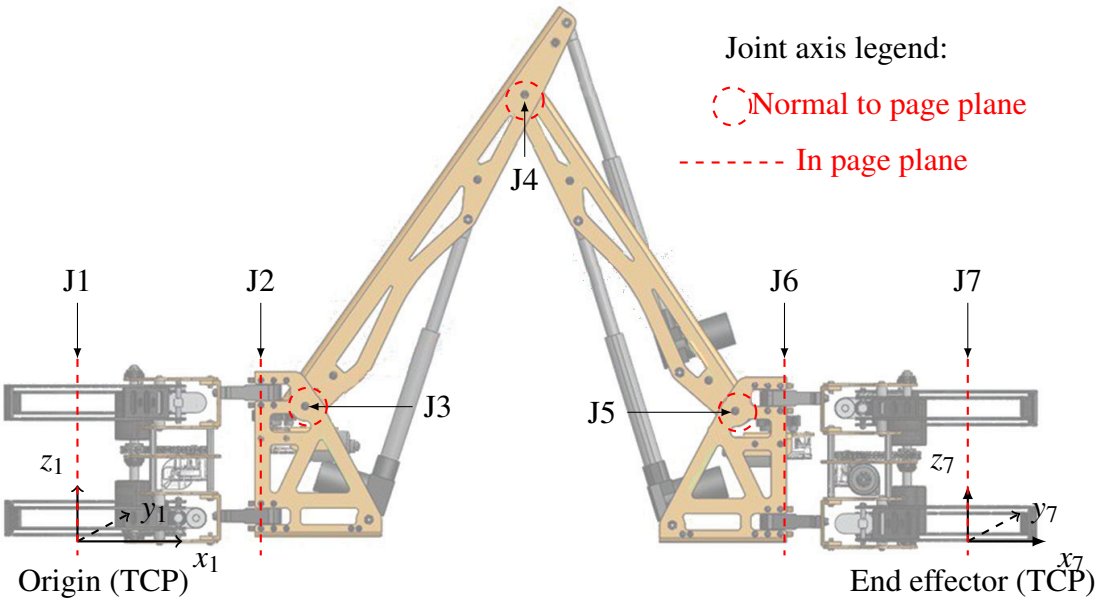


Fig. 4.1 CAD model of machine indicating joint axes

To form the kinematic model, the Denavit-Hartenberg (DH) [96, 97, 98, 99] system was applied to the geometric model identified previously. The system begins by attaching a coordinate frame to each link, located at its joint. The transformation between adjoining link coordinate frames and can then be described by a transformation matrix, defined in (Eq. (4.3)) [99].

$$A_j^{j-1} = \begin{bmatrix} \cos(\theta_j) & -\sin(\theta_j)\cdot\cos(\alpha_j) & \sin(\theta_j)\cdot\sin(\alpha_j) & a_j\cdot\cos(\theta_j) \\ \sin(\theta_j) & \cos(\theta_j)\cdot\cos(\alpha_j) & -\cos(\theta_j)\cdot\sin(\alpha_j) & a_j\cdot\sin(\theta_j) \\ 0 & \sin(\alpha_j) & \cos(\alpha_j) & d_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

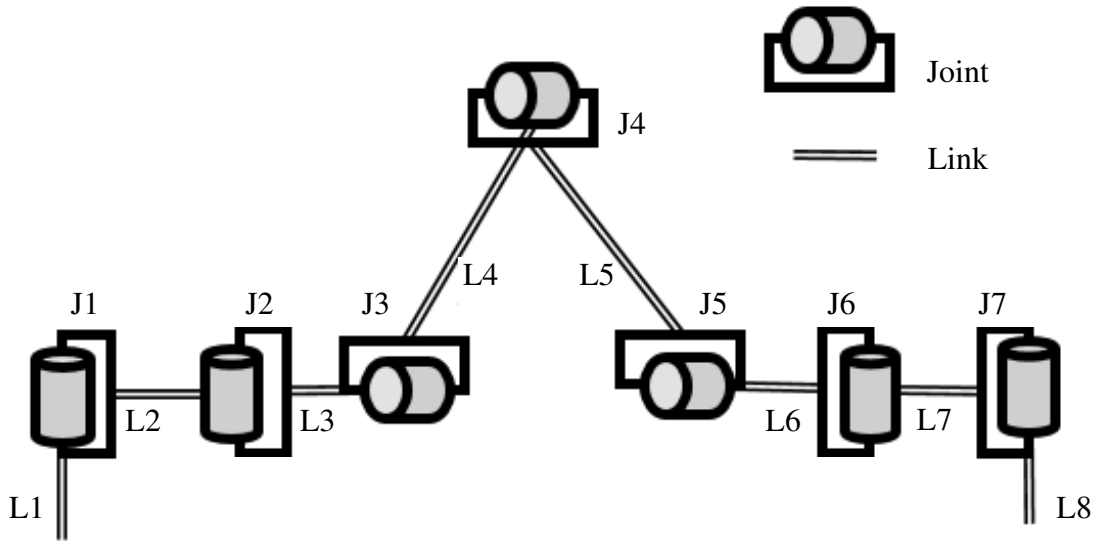


Fig. 4.2 Kinematic model of the machine

Table 4.1 Denavit-Hartenberg parameters for the machine from Gripper 1 to Gripper 2

Link	Name	Offset d (m)	Length a (m)	Twist α (°)
1	Base	0.296	0	0
2 & 7	Gripper 1 & 2	0	0.285	0
3	Wrist 1	-0.025	0.073	90
4 & 5	Arm 1 & 2	0	0.750	0
6	Wrist 2	0.025	0.073	-90
8	End effector	-0.296	0	0

The transformation between any two coordinate frames (from a to b) can then be described by the product of transformation matrices between all adjacent links that span the two frames (Eq. (4.4)). The DH parameters are listed in Table 4.1, with the joint values defined in Table 4.2. A full methodology for applying the Denavit-Hartenberg convention to an arbitrary machine is described by Spong, Hutchinson and Vidyasagar in [98]. As the machine is kinematicly symmetric, this model can be used when either Gripper 1 or Gripper 2 is fixed.

$$T_b^a = \prod_{i=a}^{b} A_{i+1}^i(q_i), b > a \tag{4.4}$$

The pose, $\mathbf{q}$, of the machine is defined as the vector of all joint angles (q_i) as in Eq. (4.5).

$$\mathbf{q} = [q_1, q_2, q_3, q_4, q_5, q_6, q_7] \tag{4.5}$$

Table 4.2 Joint angle specifications

Joint	Adjacent links	Domain (°)	Range (°)
1 (Q1)	1, 2	[0, 0]	0
2 (Q2)	2, 3	[-90.0, 90.0]	180.0
3 (Q3)	3, 4	[21.4, 76.4]	55.0
4 (Q4)	4, 5	[-130.7, -34.1]	96.6
5 (Q5)	5, 6	[21.4, 76.4]	55.0
6 (Q6)	6, 7	[-90.0, 90.0]	180.0
7 (Q7)	7, 8	[0, 0]	0

To investigate the boundaries of the workspace in the XZ plane, the joints causing rotation about the global Z axis are locked (Roller and Wrist joints). Therefore joint angles q_1,q_2,q_6,q_7 are set to zero, whilst q_3,q_4,q_5 are iterated over their respective joint range. Computing the forward kinematics of the resultant poses given by $\mathbf{q} = [0,0,q_3,q_4,q_5,0,0]$ then yields the workspace as shown in Fig. 4.3. From this analysis it is found that displacement along the X axis ranges from 0.5260 m to 2.2511 m. Displacement along the Z axis ranges from -0.6672 m to 1.9606 m.

The three dimensional workspace can be determined, as shown in Fig. 4.4, by extending the workspace of Fig. 4.3. The workspace is extended from $\mathbf{q} = [0,0,q_3,q_4,q_5,0,0]$ to $\mathbf{q} = [q_1,q_2,q_3,q_4,q_5,q_6,q_7]$ by including the joints J1,2,6 and 7. Iterating over possible joint angles for the new pose, the 3-D workspace is calculated using the forward kinematics. This indicates that the range on Y displacement is ± 1.9638 m, the maximum distance the next tree can be from the base gripper.

The manipulability of the end effector can then be investigated, following the identification of the workspace above and in Fig. 4.3. The Yoshikawa Manipulability Index [100] is used to characterize the manipulability. This is found empirically by iterating over possible joint angles, as in the calculation of Fig. 4.3, and calculating the index at the resultant pose ($\mathbf{q}$). This is shown in Fig. 4.3 for the XZ plane. The maximum manipulability was found to have a value of 3.16 which occurs at the pose $\mathbf{q} = [0.00, 0.00, 0.00, 45.87, -66.31, 21.42, 0.00, 0.00]$. By the forward kinematics, this pose corresponds to an end effector position of $[x,y,z] = [1.95, -0.03, 0.26]$. Hence at this pose the end effector is in its optimal pose for grasping a tree, allowing it to achieve arbitrary position and orientation with least effort.

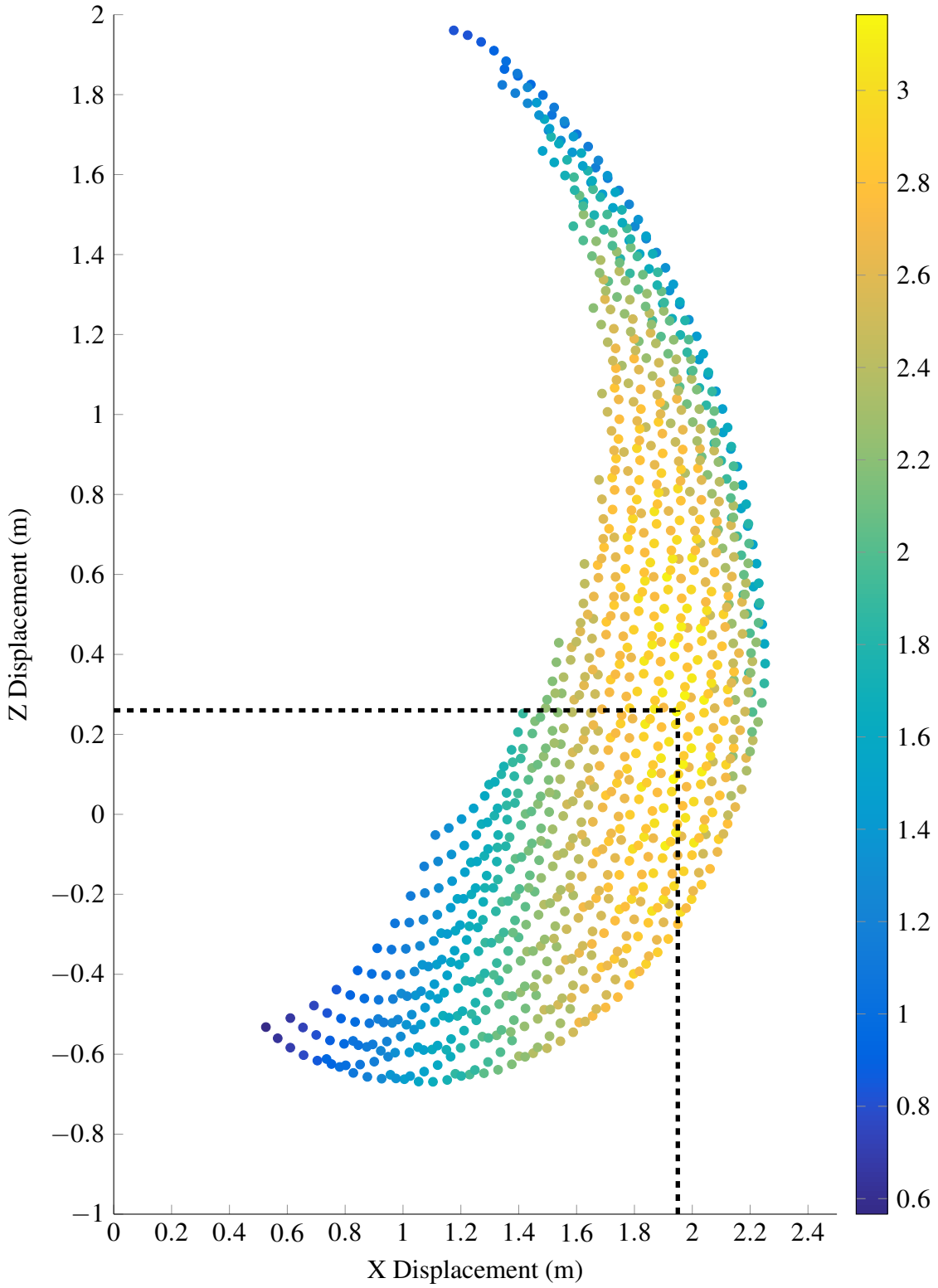


Fig. 4.3 Workspace of the end effector in the XZ plane, colored by the Yoshikawa manipulability measure (Yellow = Maximum, Blue = Minimum). The point with maximum manipulability lies at the intersection of the lines (x=1.95, z=0.26).

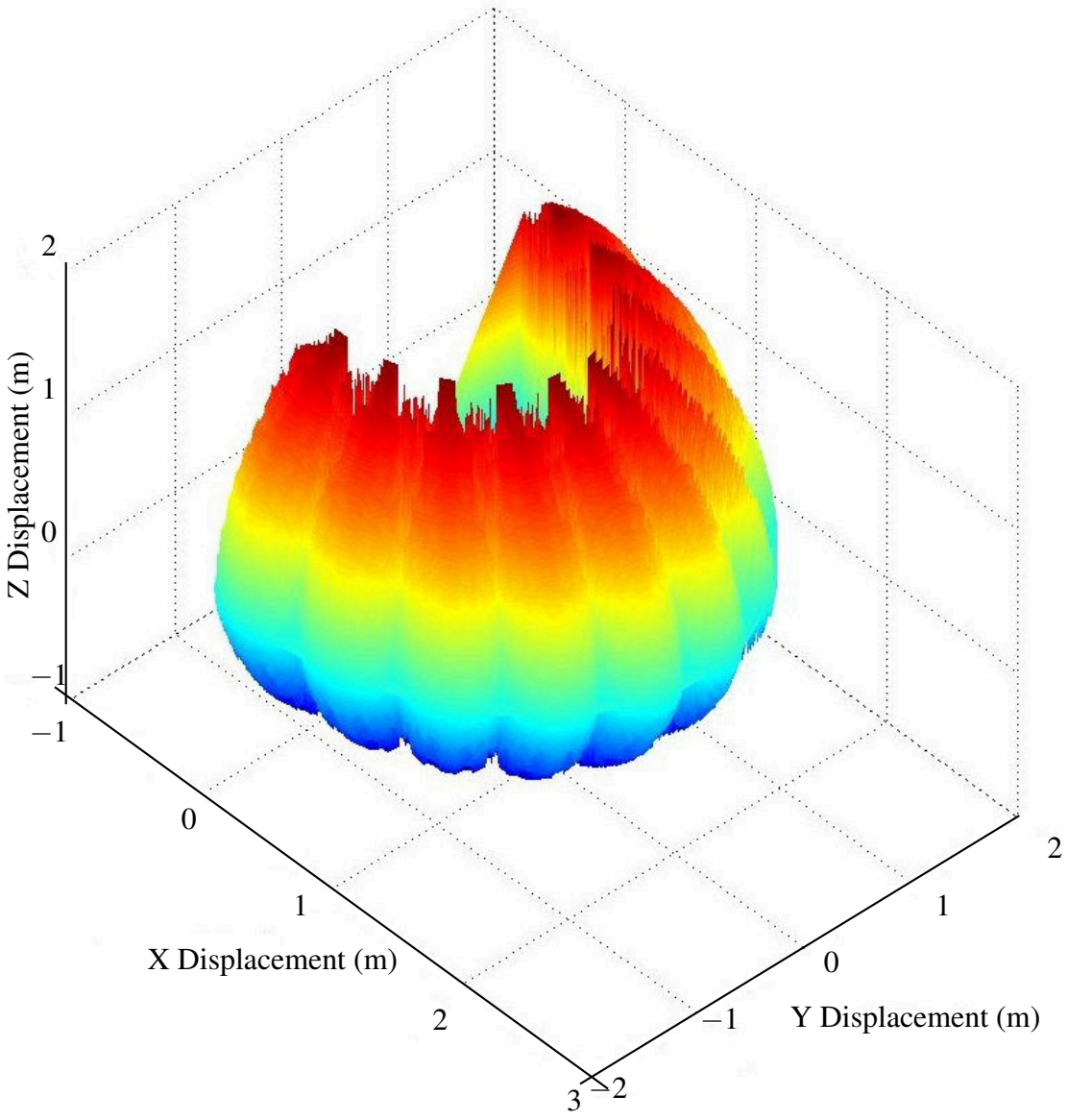


Fig. 4.4 3D view of the end effector workspace. Colour represents Z displacement (Red = Maximum, Blue = Minimum).

4.3 Machine loading characterization

Based upon the kinematic model developed previously, the dynamics of the machine can be determined. The Newton-Euler method [98] is used to determine the accelerations, forces and torques. Briefly, the Newton-Euler formulation is applied to the machine by starting at the base and working towards the end effector, the forward recursion calculates the velocity and acceleration of each frame. Having obtained the requisite velocities and accelerations, the forces and torques can then be calculated in the backward recursion working from the end effector towards the base. This model assumes that the machine can be modeled as an open kinematic chain [97, 101], associating the linear actuator mass with the parent link rather than modeling them as independent links.

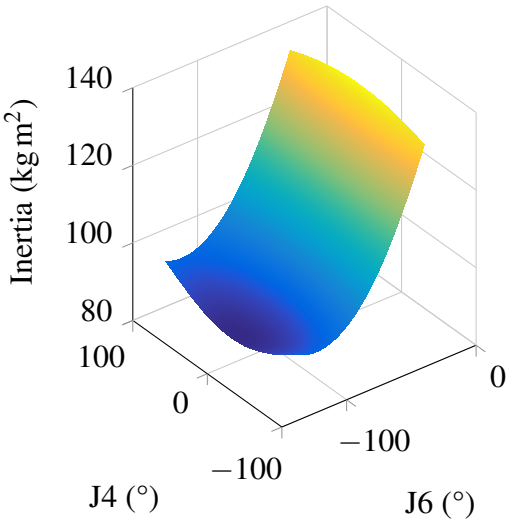
4.3.1 Inertia

Inertia describes the object's resistance to the change of motion [102]. However, "the inertia that the motor experiences is a function of the configuration of the outward links" [99] since inertia is related to the distribution of mass, m, by the square of the distance r. Therefore the inertia seen at the i^{th} joint, I_i, can be described by Eq. (4.6) where the summation of the individual link inertias' ($m_j r_j^2$) is calculated for links kinematically forward ($j \geq i$) of the joint (J_i) to be considered.

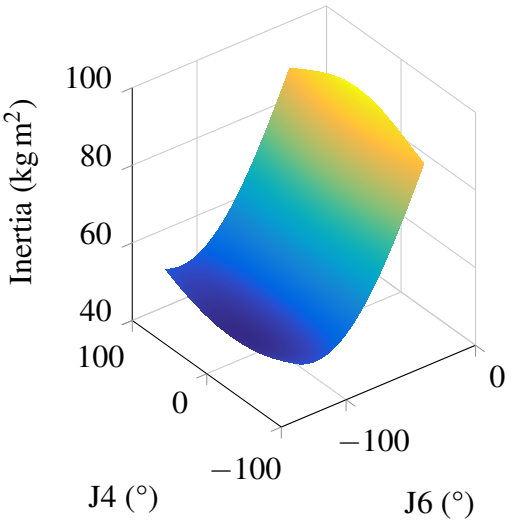
$$I_i = \sum_j m_j r_j^2 \text{ for } j \geq i \tag{4.6}$$

In order to establish the relationship of machine poses on the joint inertias', an analysis into the inertia is conducted at each joint (Fig. 4.5). At each joint, the mass contributing to the inertia is that of the outward links. The inertia at the final joint consists purely of the mass belonging to the end effector, whereas the inertia as the base joint consists of the masses of every link. A review of the inertia at all joints is conducted, the results from two joints are presented here; J1 (Base) and J3 (Arm 1). J1 is the base joint and so supports the entire machine, therefore it is important that the inertia seen at this joint be carefully considered. J3 is the first joint that operates in the XZ plane. To collect data, the 'home' pose is defined as $\mathbf{q} = [0,0,0,30,-60,30,0,0]$, as shown in Fig. 4.1. From this home pose, each joint is moved through its range to consider the individual impact of each joint.

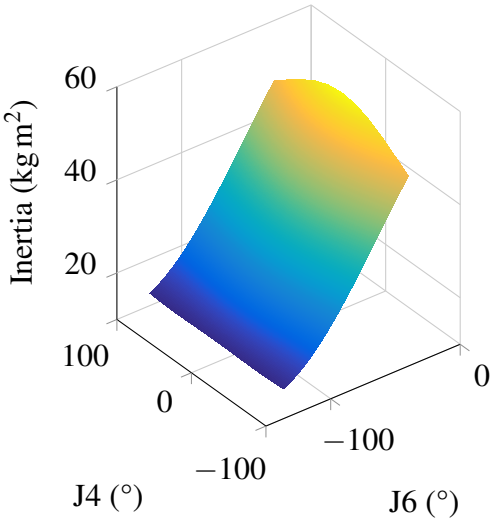
The inertia at the base joint changes with pose as shown in Fig. 4.6. It is seen that J3 (Arm 1) (Fig. 4.6a) most significantly effects the inertia that is seen by J1. Inertia ranges from



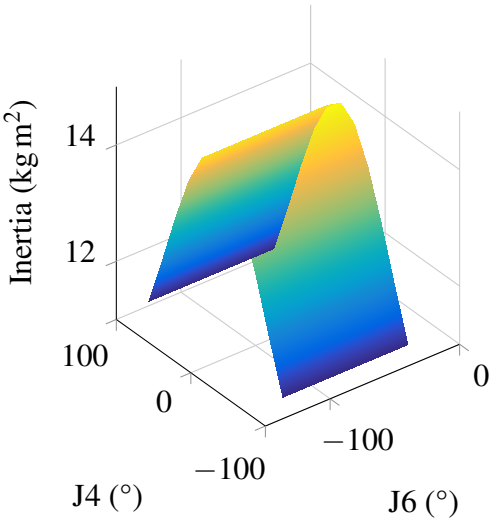
(a) Inertia at Joint 1 & 2



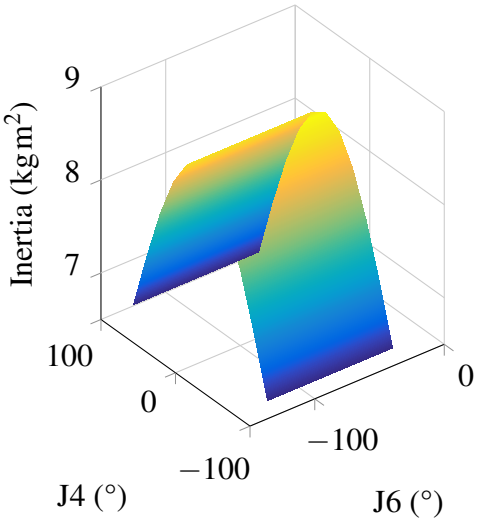
(b) Inertia at Joint 3 & 4



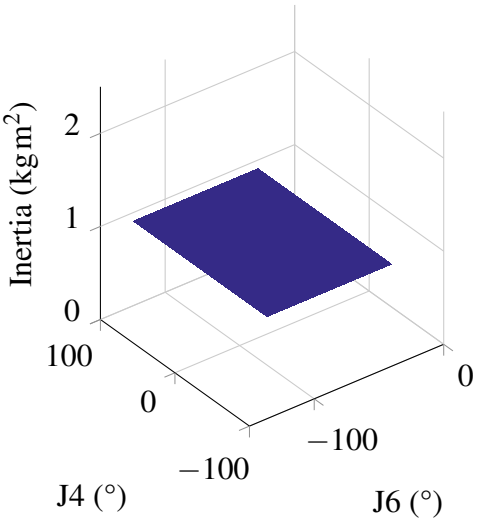
(c) Inertia at Joint 5



(d) Inertia at Joint 6



(e) Inertia at Joint 7



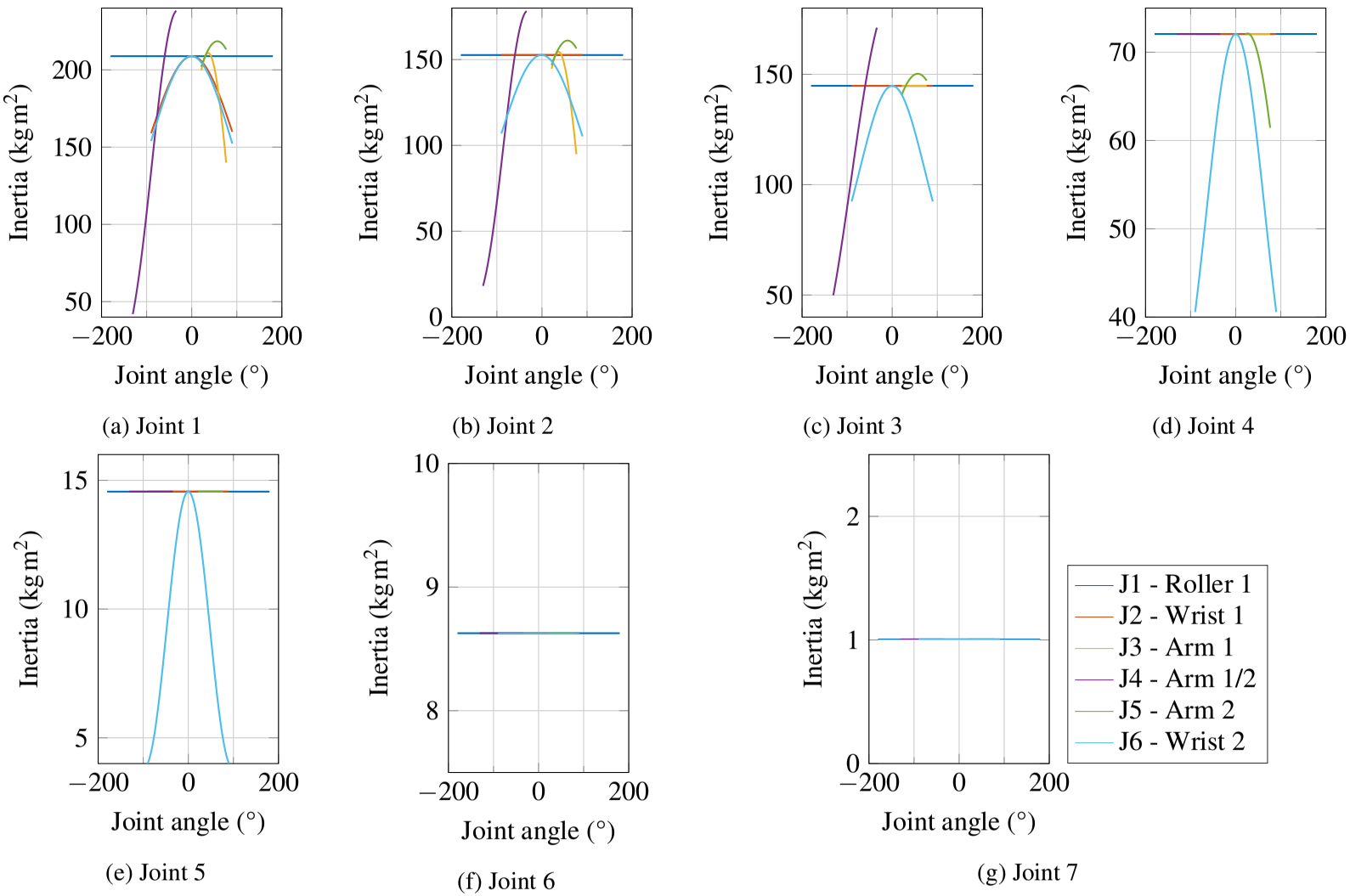
(f) Inertia at Joint 8

Fig. 4.5 Machine inertia at each joint

36.09 kg m^2 to 142 kg m^2 (δ=105.91 kg m^2) due to J3, whereas the second most influential joint, J4, causes inertia to change from 78.8 kg m^2 to 129.6 kg m^2 (δ=49.8 kg m^2). J4 has significant impact on the distribution of mass about the axis of J1 and thus the inertia due to (Eq. (4.6)). This is due to J4 having the largest joint range (Table 4.2) in the XZ plane as well as the joints position in the kinematic chain. Therefore to minimize the inertia seen by J1 it is preferable to reduce the joint angle as much as possible prior to rotation of J1 for minimum effort. The same conclusions are also arrived at for J2 (Wrist 1) as this joint operates in the same plane at J1 and is next in the kinematic chain.

The inertia at J3 is investigated as this joint is the first in the kinematic chain to operate in the XZ plane. The inertia seen from this joint is shown in Fig. 4.6c. In this case, the most influential joint on J3 is J4 which is responsible for moving all links outward from the first arm link, L4. Figure 4.7 shows the effect on inertia for varying q_2, q_3. Whilst in the home pose, operation of J4 over its complete range changed the inertia seen at J3 from 100.2 kg m^2 to 30.9 kg m^2 (δ=69.3 kg m^2). The impact of the second most influential joint (J6) shows that to minimise the inertia, the free gripper should be rotated to $q_6 = \pm 90$ deg which can generally be considered to be facing away from the target. Inertia at this joint as a function of q_3, q_4 is shown in Fig. 4.8. This is a counter intuitive result when considering how humans normally grasp an object. Generally it is desired to direct the gripper (e.g. hand) towards, not away from, the target object when reaching to grab it. The machine implementation can be adjusted such that the gripper is rotated 90° to the wrist as to minimise the inertia.

Note that due to the symmetry of the arm, when Gripper 2 is the base joint in the kinematic chain, similar conclusions are arrived at for the opposite joint in the pair: (J1, J7), (J2, J6), (J3, J5), however with lower inertia due to a tighter distribution of mass as two arm actuators are closer to the base. This can be found with the adjusted kinematic chain to reflect the changes in order.



(a) Joint 1 (b) Joint 2 (c) Joint 3 (d) Joint 4

(e) Joint 5 (f) Joint 6 (g) Joint 7

Fig. 4.6 Inertia at joints 1-8

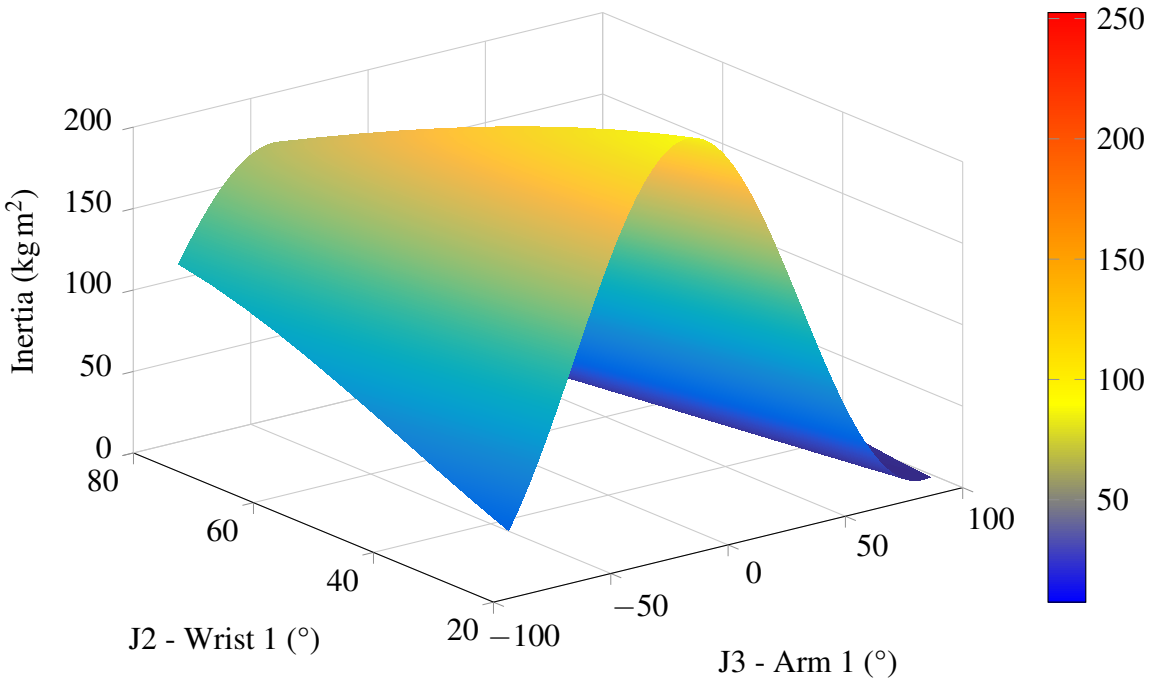


Fig. 4.7 Inertia seen at Joint 1 for varying pose of Joint 2 and Joint 3

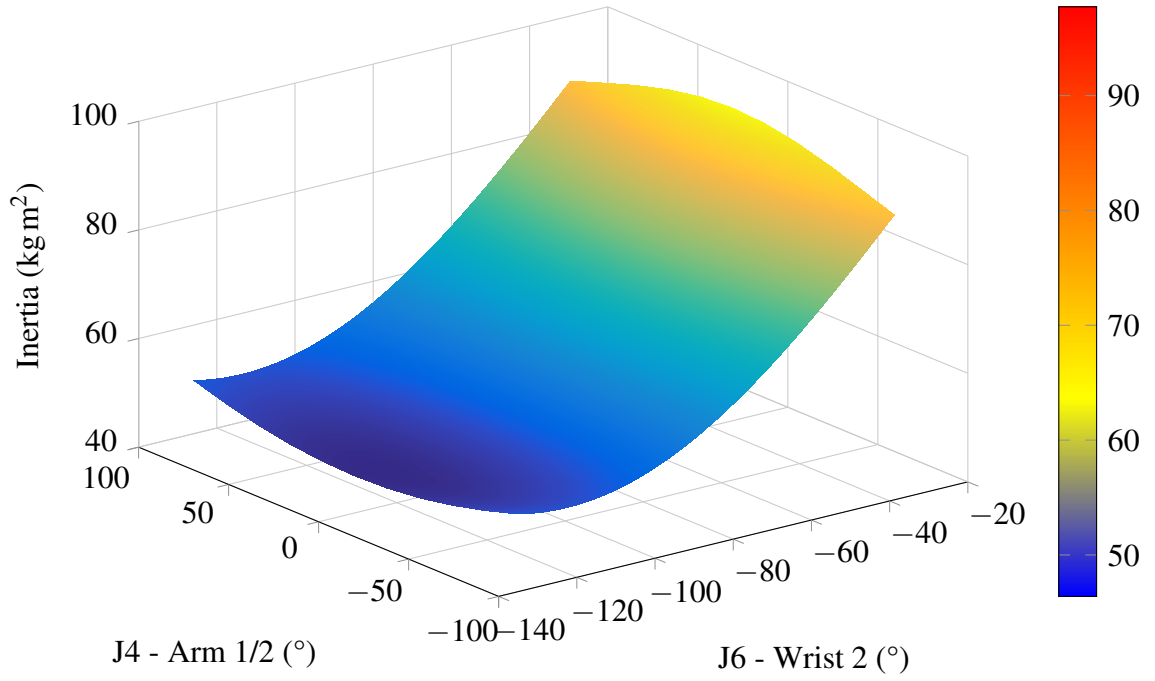


Fig. 4.8 Inertia seen at Joint 3 for varying pose of Joint 2 & 3

4.3.2 Joint torques

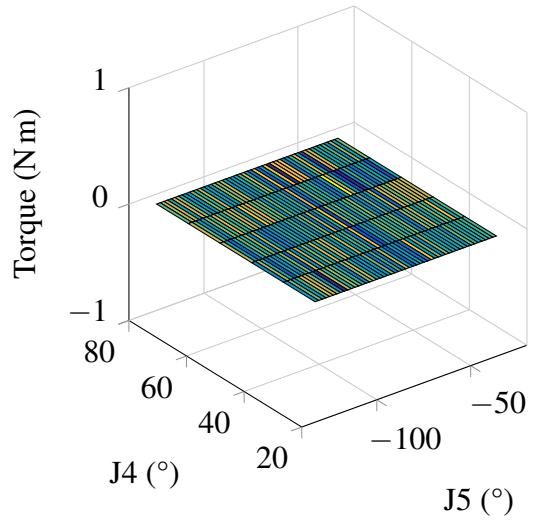
In order to determine if the actuators have sufficient static loading capacity, the effect of self-loading caused by the machine's mass and gravity is considered (Fig. 4.9). The actuators must have a sufficiently high static loading capacity to hold to the mechanism against gravity whilst also having sufficiently high torque capacity to rotate the joint through its range of motion for the required acceleration. One type of linear actuator for the three joints J3, J4, J5 as well as one type of actuator for the rotational joints J2 and J6. Evaluation of joint torque for the maximum loading case in each group of joints will be indicative of the performance at the joint with lower loading at other joints and therefore the actuator will be specified by the worst case scenario.

As linear actuators are used to achieve the rotation in J3, 4, and 5, the available joint torque is not linear. The joint torque provided by the linear actuator can be found by the cosine rule [103], the joint angle and the geometric properties of each joint. The tangential force component that creates the torque, applied by the actuator, can be calculated using the applied actuator force f. The linear actuators used in this system are PA-17s from Progressive Automation [104] which can supply a maximum force of 7063.2 N. The joint torques resulting from the PA-17 actuators are calculated, Fig. 4.10 shows their output with respect to their joint angle.

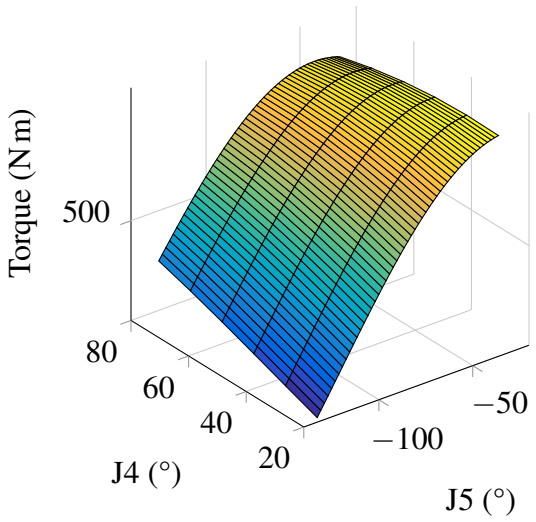
The joint torques created by the effect of gravity on the self-load of the machine are calculated with the robot positioned in the home position. A dynamic model based upon the DH kinematic model forms the basis for calculating joint torques. This is formed by adding the link masses and moments. Calculating the torques created by gravity and by the actuator at each position allows for a comparison between what is available and what is required, as shown in Figs. 4.11 and 4.12 for joints 3 and 4 respectively. The results show that whilst in the home position, the worst case loading for Joint 3 is up to 63.24% of its maximum static loading and for Joint 4 as much as 90.12%.

Table 4.3 Upper bounds of acceleration of joints J2 and J6 when at home position, changing (q3, q4) and (q5, q7) respectively.

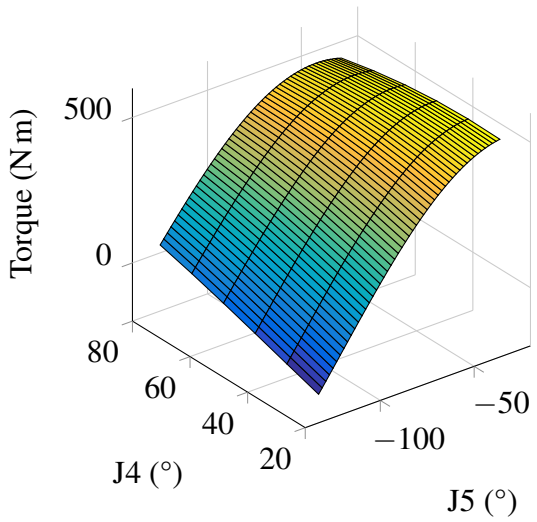
	Maximum instantaneous acceleration $\mathrm{rad\,s^{-2}}$	
	Lower bound	Upper bound
Joint 2	0.36	4.79
Joint 6	1.89	23.97



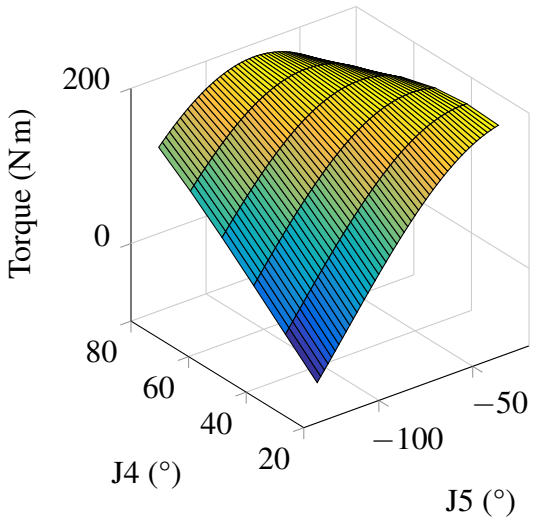
(a) Joint 1, 2 & 3



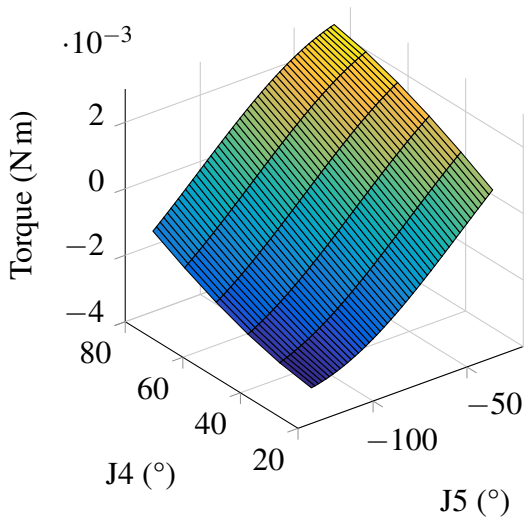
(b) Joint 4



(c) Joint 5

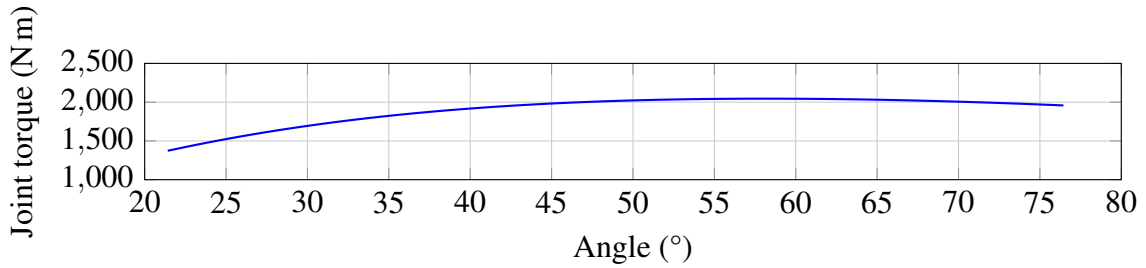


(d) Joint 6

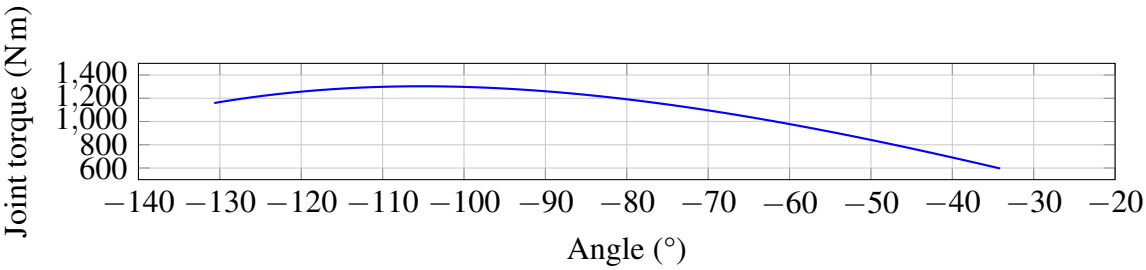


(e) Joint 7 & 8

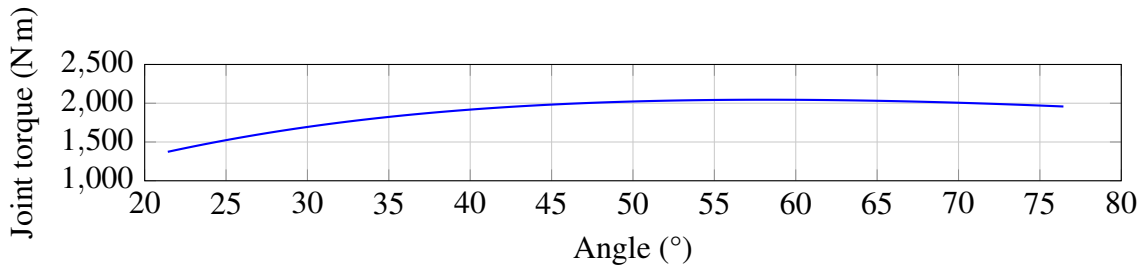
Fig. 4.9 Torque at joints 1-8



(a) J3 - Arm 1



(b) J4 - Arm 1/2



(c) J5 - Arm 2

Fig. 4.10 Maximum joint torque capacity supplied by actuator for Joints 3, 4, and 5 as a function of their respective joint angle

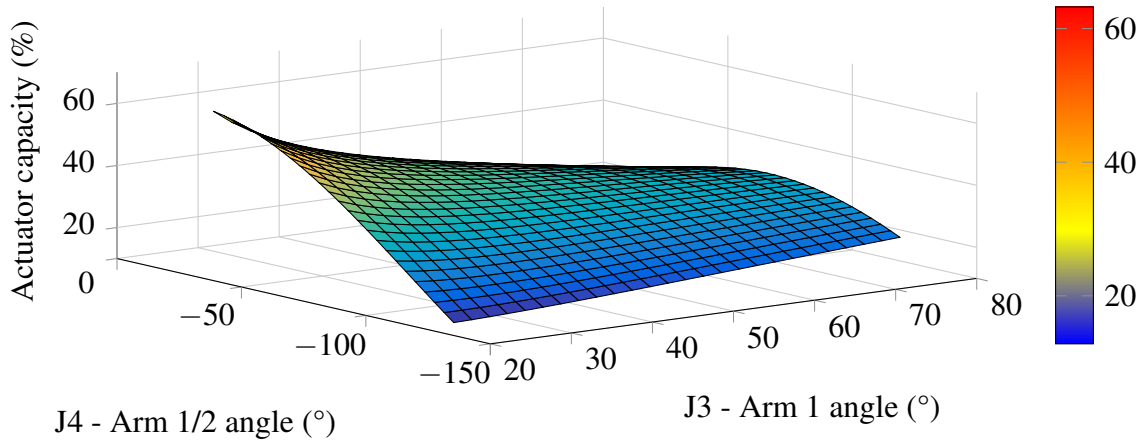


Fig. 4.11 Utilised actuator capacity at Joint 3 due to the gravitaional self-load at the home pose with varied Joint 3 & 4

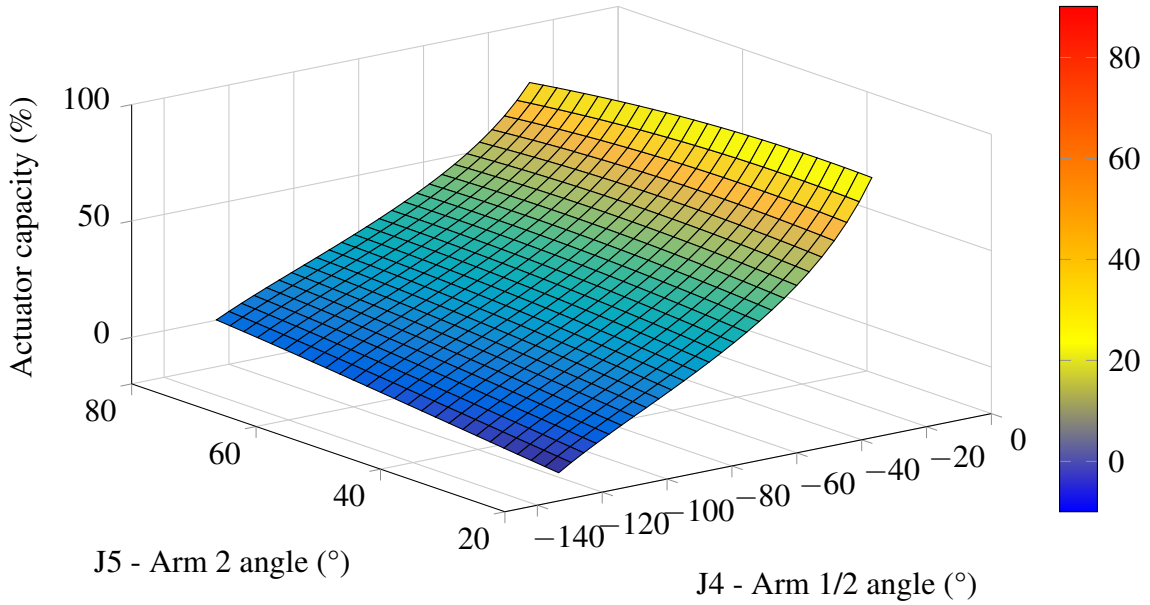


Fig. 4.12 Utilised actuator capacity at Joint 4 due to the gravitational self-load at the home pose with varied Joint 4 & 5

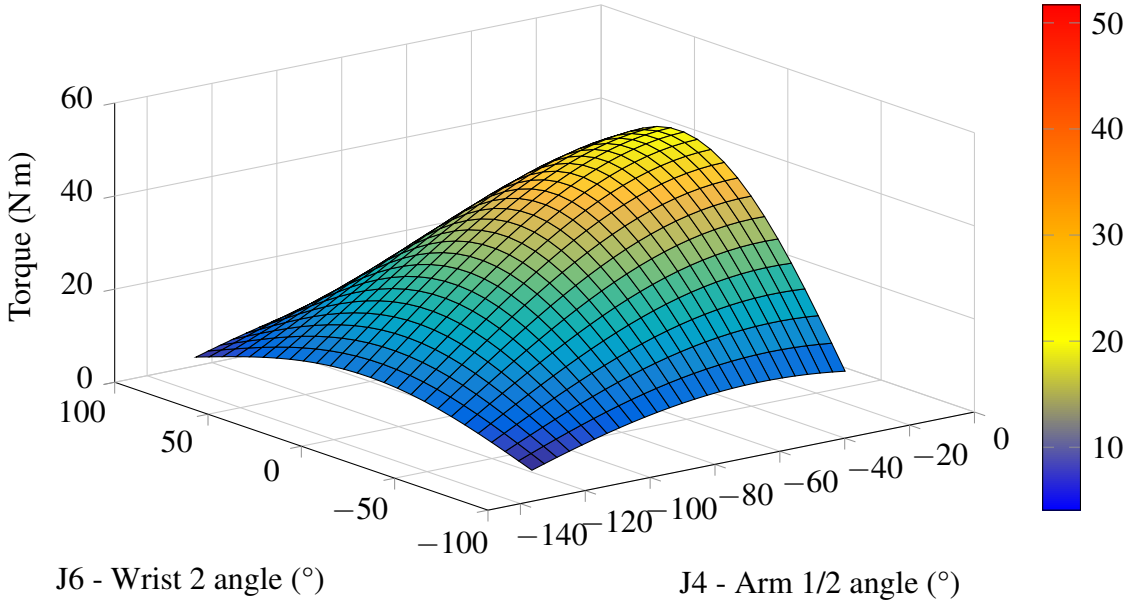


Fig. 4.13 Torque at Joint 6 - Wrist 2

4.3.3 Joint acceleration

The torque required to achieve a desired acceleration can be calculated through the relationship that the torque is the product of inertia and acceleration. Performance limitations regarding acceleration can then be identified. Accelerations are calculated using a loading of machine self-weight caused by gravity.

The upper bound on acceleration was determined for the rotational joints J2 and J6. These joints are actuated by a motor [105] with a stall torque of 26 N m. The shaft is connected to the joint by gearing with a ratio of 2.5:1, producing an overall torque of 65 N m.

Joint 2 can achieve an acceleration of $0.3547\,\mathrm{rad\,s^{-2}}$ for any variations of the home pose that changes in joint angles for J3, J4 can create. In the extreme case an acceleration of $4.785\,\mathrm{rad\,s^{-2}}$ can be generated. This can only occur in the pose where J3, J4 are minimized to $-130.7°$, $21.42°$ respectively, minimising inertia. As J6 has lower loading due to being near the end of the kinematic chain, the mechanical upper bound on the acceleration can be as high as $23.97\,\mathrm{rad\,s^{-2}}$. The upper bounds for acceleration of the two wrists joints have been found as per Table 4.3.

4.4 Conclusion

Models to describe the kinematics and dynamics of the brachiating tree-to-tree machine were developed. The kinematic analysis provided identification of the workspace of the end effector. Within the workspace, the Yoshikawa manipulability index was used to identify the manipulability at each point and found the optimal grasping point for the end effector. The dynamic model was used to find inertia, torque and acceleration at each joint. The relationship between pose and inertia was considered and used as a means to minimize effort by first reducing the inertia seen at the considered joint. Actuator performance was characterized by the joint torque. It was found that J4 can experience up to 90.12% of its maximum loading. Acceleration performance of the rotational joints was determined. Identification of the poses which minimize and maximize the joint speeds showed that J2 and J6 can reach at least $0.36\,\mathrm{rad\,s^{-2}}$ and $1.89\,\mathrm{rad\,s^{-2}}$ and an absolute maximum of $4.79\,\mathrm{rad\,s^{-2}}$ and $23.97\,\mathrm{rad\,s^{-2}}$ respectively.

Future work will develop a control strategy that is aware of the machine properties identified in this chapter. Poses leading to optimum actuator performance or grasp can be

used to optimize long term performance of the machine for energy efficiency or speed. The route planning of the machine though the plantation forest can benefit from the detailed model of workspace manipulability. Planning can choose the order of trees traversed so that optimal grasping occur as frequently as possible whilst also maximizing advantages in kinematic redundancy.

Chapter 5

k-Means Partitioned Space Path Planning (KPSPP) for autonomous robotic harvesting

A three-dimensional coverage path planning algorithm is proposed for discrete harvesting machines. Although prior research has developed methods for coverage planning in continuous crop fields, no such algorithm has been developed for discrete crops such as trees. The problem is formulated as a graph traversal problem and solved using graph techniques. Paths to facilitate autonomous operation are generated. A case study is formed around the novel tree-to-tree felling system developed by the University of Canterbury and Scion. This machine is being developed to maneuver through New Zealand's plantation forest to fell Pinus radiata trees on steep ($\geq 20°$) terrain. Algorithm performance is evaluated in 14 *commercial plantation forests. Results indicate that mean coverage of* 84.43 *% was achieved.*

5.1 Introduction

The current state of the art for coverage path planning, as applied to harvesting, has been limited to continuous crops such as wheat in large fields [52, 53, 55]. Currently there are no such algorithms for harvesting discrete objects such as trees, to the knowledge of the authors. Machines such as the ClimbMax [24], John Deere 909 [18] and the tree robot proposed in Chapter 3 can benefit from algorithms specifically designed for their industry and unique considerations such as discretized crops, steep terrain and coverage requirements.

A three-dimensional coverage path planning algorithm is proposed for discrete harvesting machines. Although prior research has developed methods for coverage planning in continuous crop fields, no such algorithm has been developed for discrete crops such as trees. The planning problem is formulated as a graph traversal problem and solved using graph-theory techniques. Paths to facilitate autonomous operation are generated. The novel contribution of this chapter is that of solving the two and three dimensional coverage route planning problem for a node-to-node style harvesting machine. This has applications to a wide variety of field robotics which must visit multiple discrete sites, even outside harvesting such as; search and rescue, security and scientific sampling of interest points. Optimizing the visitation to these sites can improve performance in measures such as time, vehicle lifetime and fuel consumption.

5.2 Planning methodology

For many applications in harvesting the former coverage cases [52, 53, 55] can be an ill-fitting abstraction of the route planning problem. Where the object to be harvested can be considered as discrete (for example radicchios and pine trees) as opposed to continuous (maze, sugar cane), then a more fitting abstraction might be that of a node visitation problem. Commercial plantation forest in New Zealand is generally on rough, steep terrain [6]. Trees that are planted in regular rows, develop over their 30 year lifespan to be in irregular positions and poses due to a variety of environmental factors. In this and other cases each element of the crop can be considered a node and a path between nodes as an edge. Therefore, to map the graph-theory terminology to the present case: A node represents a tree and an edge represents a path connecting two adjacent trees. The methodology considers the approach taken towards coverage path planning for the purpose of discrete harvesting. Nodes (trees) can only be visited once since upon harvesting they are removed. After visiting the node it cannot be used as a 'stepping stone' to traverse to other nodes. The methodology can be broken into four constituent parts (Fig. 5.1), namely: A. Edge identification, where inter-tree path options are identified; B. Graph representation, where a graph model is created. C. Partitioning, to reduce the computational complexity; D. Partitioned Space Path Planning using k-Means, the path planning operation.

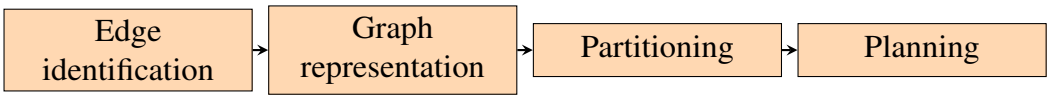


Fig. 5.1 Methodology overview flow chart

5.2.1 Edge identification

Input to the algorithm is the data collected from the forest inventory, including the latitude, longitude and altitude for each of the n trees. This tree data might be obtained, for example, by aerial LiDAR or terrestrial GPS. This work uses provided forest maps which have been generated by finding the crown location and inferring the trunk location at ground level from Pont et al. [76]. From these known tree locations the algorithm must derive the graph structure. The graph structure abstraction sets the basis for the coverage path.

Whilst nodes are readily determined from the tree location data by the one-to-one relationship of trees to nodes, graph edges need to be defined. In the forest there are no predefined paths between trees, therefore inter-tree paths are defined to be the edges of the triangles generated in the Delaunay Triangulation [90]. By using the Delaunay Triangulation, nodes are connected in the nearest neighbor fashion. The triangulation also contains the minimum spanning tree and convex hull of the set. Points on the convex hull are the most likely start points for a harvesting machine due to their convenient location on the periphery of the harvest area. From the latitude and longitude of each node the triangulation is computed. By decomposing the resultant triangles into its edges, ε paths between nodes are established.

The Delaunay Triangulation determines edges as shown in Fig. 5.2. Tree locations, shown as circles, are fed into the triangulation algorithm. All edges from the triangulation are shown in dashed lines. Due to the limited workspace of the machine however, not all these edges can be traversed. A filtering step removes edges too long or too short for the machine based upon the 3D Euclidean distance for each edge. The solid lines indicate the traversable paths, these are the paths used in the planning process. These solid lines form the set of edges ε.

5.2.2 Graph representation

To form an efficient data structure to contain the graph connectivity and edge costs, an adjacency matrix $\boldsymbol{\alpha}$ is formed defined by Eq. (5.1). The edge, ε_k, joins two nodes, n_i to n_j. Each element in the adjacency matrix represents the cost $C(\varepsilon_k)$ of traveling along the edge.

$$\boldsymbol{\alpha}_{(i,j)} = \begin{cases} 0 & i = j \\ C(\varepsilon_k) & i \neq j \end{cases} \tag{5.1}$$

The cost function is defined by the interests of the forest manager. Interests may, for example, include the number of trees harvested, volume harvested, time taken and fuel consumed. A cost function might be binary, indicating if traversal along that edge is possible.

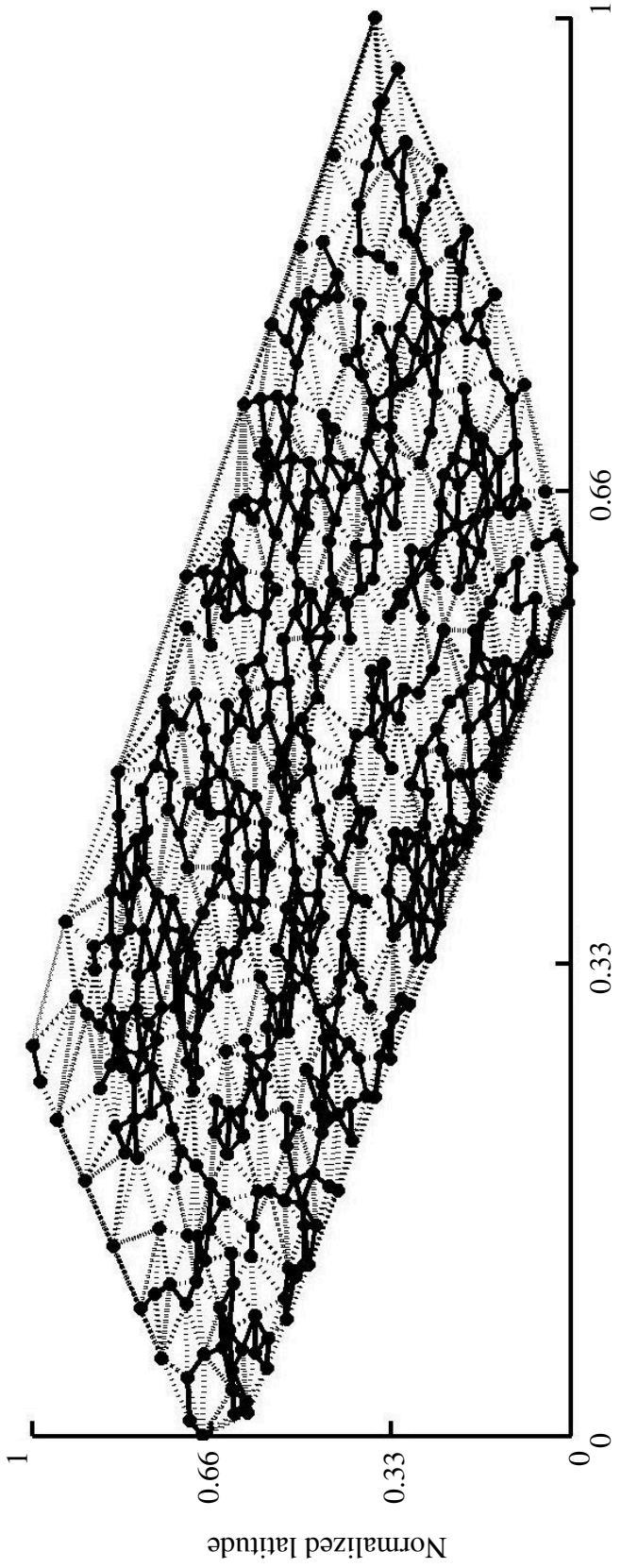


Fig. 5.2 Identified paths in forest: dashed lines indicate all paths, solid lines indicate traversable paths and nodes indicate tree locations. Actual coordinates are abstracted due to commercial sensitivity.

If an edge (ε_k) between two trees (n_i, n_j) has length d, then the edge is traversable if the length is less than or equal to the machine span (d_{span}). Similarly the cost function could be defined directly as a function of the edge length (Eq. (5.3)), as used in this work. Other examples of edge costings include, but are not limited to, vertical displacement of edge, time to travel edge or energy cost to traverse edge.

$$C(\varepsilon_k) = \begin{cases} 0 & d_{edge} > d_{span} \\ 1 & d_{edge} \leq d_{span} \end{cases} \tag{5.2}$$

$$C(\varepsilon_k) = \begin{cases} 0 & d_{edge} > d_{span} \\ d_{edge} & d_{edge} \leq d_{span} \end{cases} \tag{5.3}$$

Relevant elements represent traversable edges which have non-zero values. Similarly, zero valued elements are defined *irrelevant* due to having a non-traversable property. If the ratio of relevant to irrelevant elements is considered small then the matrix is said to be sparse. Tewarson [106] indicates that in practice if the matrix has in the order of n nonzero elements, for large n, it is considered sparse. It is inefficient to store edge costs of a low connected graph in an n by n matrix. An alternative is to use the sparse class of matrices that Matlab offers. MathWorks [67] give an example: Consider the identity matrix for n=10 000. In the standard matrix format this would require provision for 100×10^6 values. If double precision floating point numbers are used, this would occupy 800 MiB. In sparse format, this matrix would require approximately 0.24 MiB. In order to use the sparse matrix representation in Matlab efficiently, zero must be used as the edge cost for non-existent edges rather than infinity. This is because sparse matrices only store the values of non-zero elements rather than every element in standard matrices – leading to significant size reduction for large, low density matrices. Fig. 5.3 shows how the size of both matrix forms change with size.

The target machine can traverse an edge in either direction, from one tree to another in either order. Graph edges are therefore bidirectional due to this traversal property. This leads to an adjacency matrix that is symmetric along its main diagonal ($\boldsymbol{\alpha}_{(i,j)} = \boldsymbol{\alpha}_{(j,i)}$).Due to this symmetry, the upper triangular half of the matrix is sufficient to fully define $\boldsymbol{\alpha}$. The full matrix is formed by an OR operation between the complete upper half and its transpose: $\boldsymbol{\alpha} = OR(\boldsymbol{\beta}, \hat{\boldsymbol{\beta}})$. Although the full matrix can be easily computed, the storage size of the matrix can be halved at this point by using only the upper triangular area.

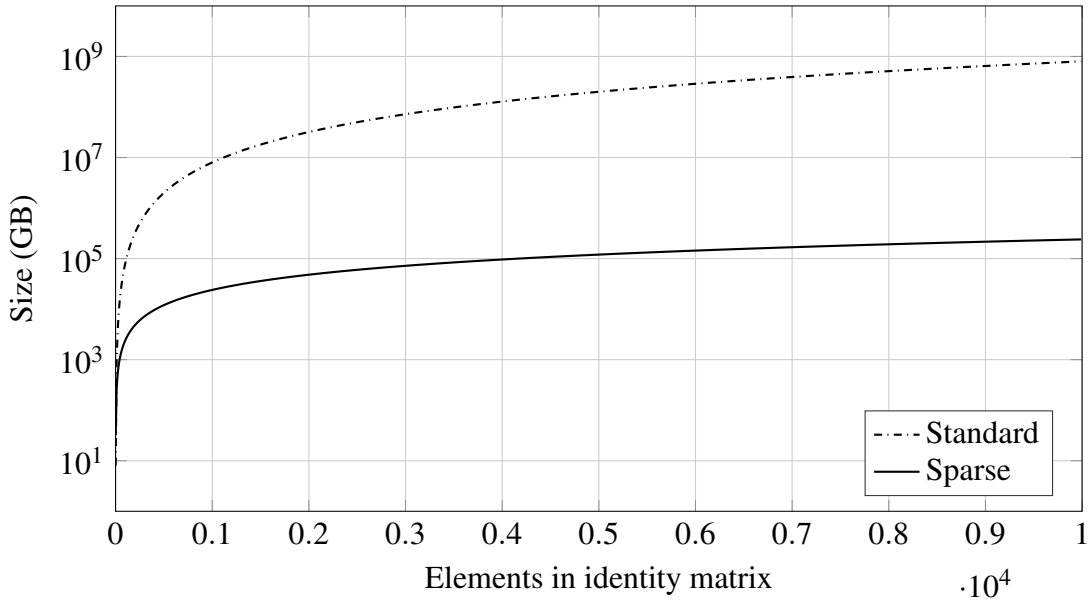


Fig. 5.3 Comparison of the storage size of identity matrices in standard and sparse forms

5.2.3 Partitioning

In the case of harvesting, trees can be visited exactly once and so the ideal case occurs when all nodes are visited. For a graph of n nodes, the longest route through the graph visits exactly n nodes. The maximum number of complete paths through the nodes is known a priori, given by the permutations. Therefore, for n nodes there are no more than $n!$ complete routes. This does not include the partial coverage paths which do not visit every node. A matrix used to store routes through the graph would have dimensions of $(n!, n)$, or larger if partial coverage routes are included. The storage size of this matrix exponentially increases with n and is therefore impractical to maintain for even small n. The amount of storage required (S) is given by Eq. (5.4) when storing 64 bit numbers.

$$S = 8n \cdot n! \,\text{Bytes} \tag{5.4}$$

The rapid growth in size is demonstrated in Fig. 5.4 where, for example, at n=11, 3.3 GB of storage is required. For this reason an upper limit on μ is created. Therefore it is necessary to partition the dataset so that no more than μ nodes are planned at any one time.

A recursive partitioning strategy is developed. This method reduces the memory requirements of the algorithm by planning only subsets of the total problem and appending the subgroup path to the global path to form the global solution. Section 5.2.3 describes the

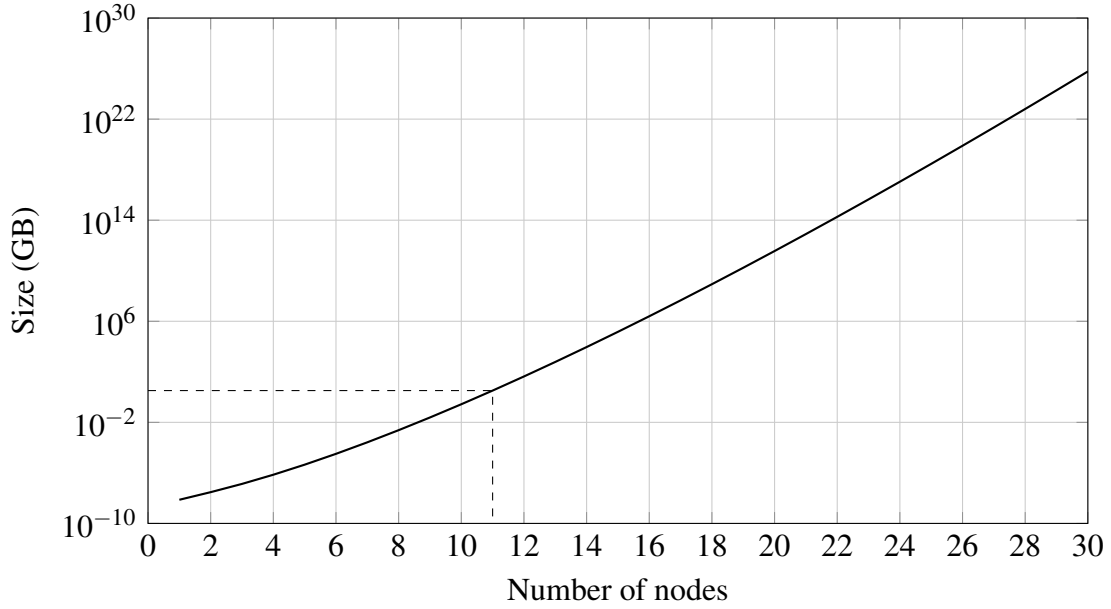


Fig. 5.4 Growth of adjacency matrix storage size with node quantity, $n = 11$ highlighted

algorithm used.

Groups of nodes exceeding μ members must be partitioned to form smaller subsets. Fig. 5.5 illustrates an example forest with 31 trees and maximum planning capacity, μ, of 4. Initially the forest has 31 trees which exceed μ. The group is partitioned therefore into μ subsets (indicated as yellow, green, blue and red) to form the first level of recursion. The yellow, green, blue and read subgroups contain 9, 7, 9 and 7 nodes respectively. This partition of four subsets can now be planned to obtain the order in which they will be traversed; the order obtained might be yellow, green, blue then red. As each subgroup contains more than μ members, they must in turn be partitioned in order to complete the plan in finer resolution. The yellow subset, containing 9 nodes, is divided into three subsets; Yellow, blue and green forming the second level of recursion. As this subset contains less than μ members, the path can be planned and results in the order yellow, green then blue. At the third level of recursion, each subset now contains less than μ members and so the path through these trees can be obtained directly. Therefore the path through the trees of the yellow, level one, subset is discovered and forms the beginning of the global path. The algorithm now returns to level one and continues to the next subset according to the plan (green). Again the algorithm recurses to find the route. Once the route through the trees is achieved for this subset, it is appended to the global path. This repeats for the remaining level one subsets (blue and red)

Algorithm 1 Recursive partition and plan function

1. start point, end point = Defined from parent level plan.
2. nodes = Array of locations to visit
3. If (length(nodes) > μ)
   (a) Partition nodes to form subsets
   (b) Plan path which tours through subsets centroids
   (c) For each subset in path
      i. Recurse with nodes set to data from subset
   (d) End
4. Else
   (a) Plan path through nodes
5. End

until the endpoint it reached.

The partitioning of a group is created using the k-Means algorithm. Due to the limitation on the number of nodes, no more than μ subsets can be created at any level. The algorithm starts by randomly assigning each data point to one of the μ groups. Iteratively, nodes are reassigned to groups so as to minimize the cost of the grouping. The cost based upon the squared Euclidean distance. Once the cost does not change significantly from iteration to iteration a local minima has been found and the grouping complete.

The adjacency matrix summarizing the graph connectivity for any level can be quickly reformed from the global adjacency matrix computed initially. The path costs for edges not starting in the current set ($\mathbf{S}_i$) are set to zero (Eq. (5.5)), the ignore case, whilst the edges not finishing in the next set ($\mathbf{S}_{i+1}$) are also set to zero (Eq. (5.6)).

$$\boldsymbol{\alpha}_{(a,b)} = 0 \quad a \not\subset \mathbf{S}_i, \quad b \in [0,n] \tag{5.5}$$

$$\boldsymbol{\alpha}_{(b,a)} = 0 \quad a \not\subset \mathbf{S}_{i+1}, \quad b \in [0,n] \tag{5.6}$$

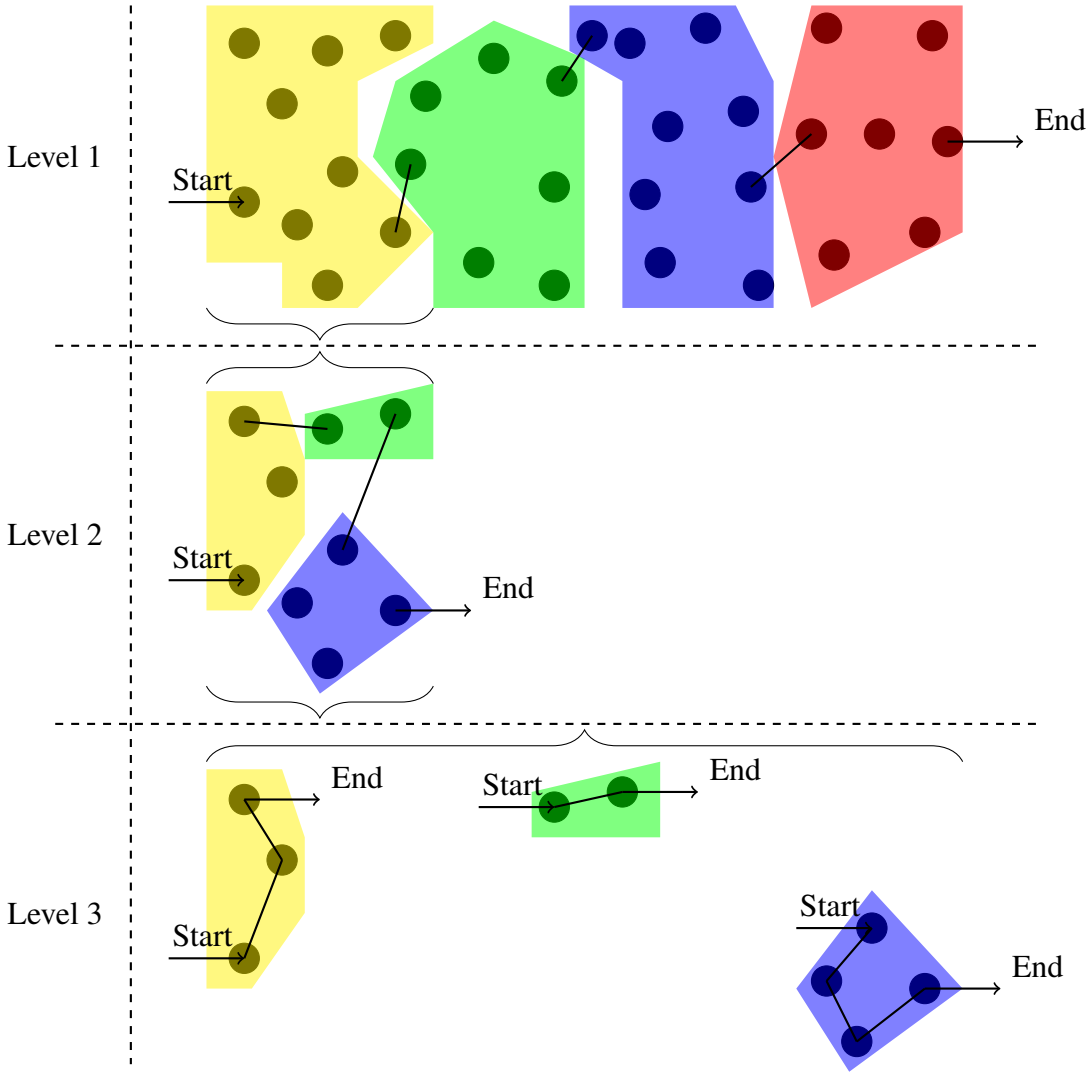


Fig. 5.5 The first three levels of recursion for the example of $\mu = 4$

5.2.4 Partitioned space path planning using k-Means

In this work it is assumed that the number of nodes visited is of a higher priority than the cost of the route. Therefore where multiple options exist for the harvesting route, the longest will be chosen. If there are multiple different routes with the same, maximum length, the lowest cost route will be chosen. This is justified by the harvesting goal, in which the goal is to harvest the maximum number of trees. Application of existing algorithms that are limited computationally by the graph size, are able to be implemented through the partitioning approach developed.

Algorithm 2 Node planning algorithm

Start (N_s) and end (N_e) node defined from parent level plan.

Initialize the set of routes, $\mathbf{r}_{(1,1)} = N_s$

Initialize candiadate route list $\mathbf{r} = \emptyset$

For each route in possible route set, $r \in \mathbf{R}$:

1. Identify adjacent nodes $\mathbf{S}_{i+1}$ where $\boldsymbol{\alpha}_{(\mathbf{S}_i,\mathbf{S}_i)} \neq 0$
2. If no adjacent nodes, $\mathbf{S}_{i+1} = \emptyset$
   Discard route r_i from route set $\mathbf{r}$.
3. Else
   (a) For each node $N \in \mathbf{S}_{i+1}$:
      i. Append N to r_i.
      ii. Clone r_i to r_{i+1}, creating new possible route.
      iii. If $N = N_e$
         A. Append r_i to list $\mathbf{R}$
         B. Discard r_i from $\mathbf{r}$, no further exploration of route
      iv. Next node, $i = i + 1$
   (b) End
4. End

End

Return results set $\mathbf{R}$

At the planning stage the adjacency matrix, start and end points are already defined. The human operator specifies the two points at which the harvesting should start and finish

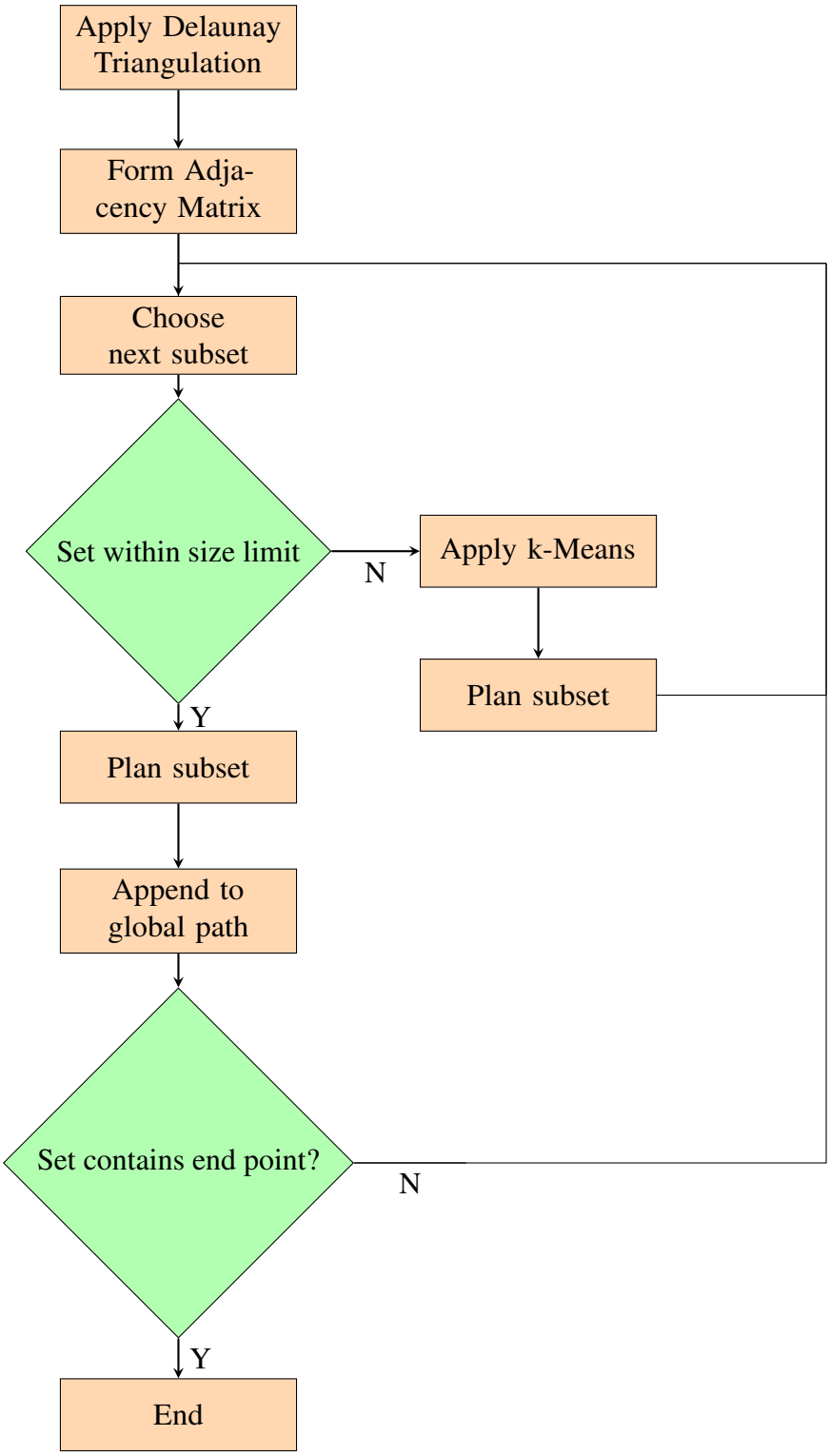


Fig. 5.6 Flow chart of algorithm

for the set as a whole. In the case of an intermediate set, the end point is chosen to be a point which has connectivity outside of the set to the following subgroup. The start node for intermediate sets is determined by choosing the point in the next set with the minimum Euclidean distance. The planning component of the algorithm is listed in Algorithm 2. Paths are explored in parallel as they are discovered. Starting from the initial node, the adjacency matrix is inspected for edges connecting the current node to others. For each edge that leads to an unvisited node, the previous path is duplicated and the next node appended to create a new path. This repeats until either the target end point has been reached or there are no nodes left to move to. The latter case occurs when there are no edges connecting the current node to an unvisited node. Routes that do not reach the end point are excluded from further analysis. This returns all possible paths in the set between the start and end point. As all the paths have varied length, only the maximal length path is chosen. If there are multiple paths with the same coverage the algorithm selects the one with the minimal cost according to the sum of the edge costs as defined by the heuristics.

5.3 Results

5.3.1 Graph formation

The Delaunay Triangulation was computed for a set of 486 nodes (Fig. 5.2). This generated 2952 distinct edges. Filtering edges due to the machine span limitation reduced edges to 1569 edges (53 % the original quantity). In the data natural subsets occur due to the geography which the machine cannot bridge. Intervention would be required to assist the machine traverse from one set to another. The natural subsets also help to reduce the planning problem. Initial this dataset had 486 nodes in one set, after the consideration of the geography there are four naturally enforced subsets of: 401, 67, 19 and 2 nodes each. These are created naturally where there are regions between which the robot span is insufficient to bridge the gap. The natural subsets occurring in Fig. 5.2 are identified in Fig. 5.7.

The occupation of the adjacency matrix for the largest set in Fig. 5.7 is shown in Fig. 5.8. The figure presents the occupied nodes of the adjacency matrix with black squares, white areas for empty areas. The square matrix has a width of 401 elements and, in this case, contains 1274 non-zero elements.

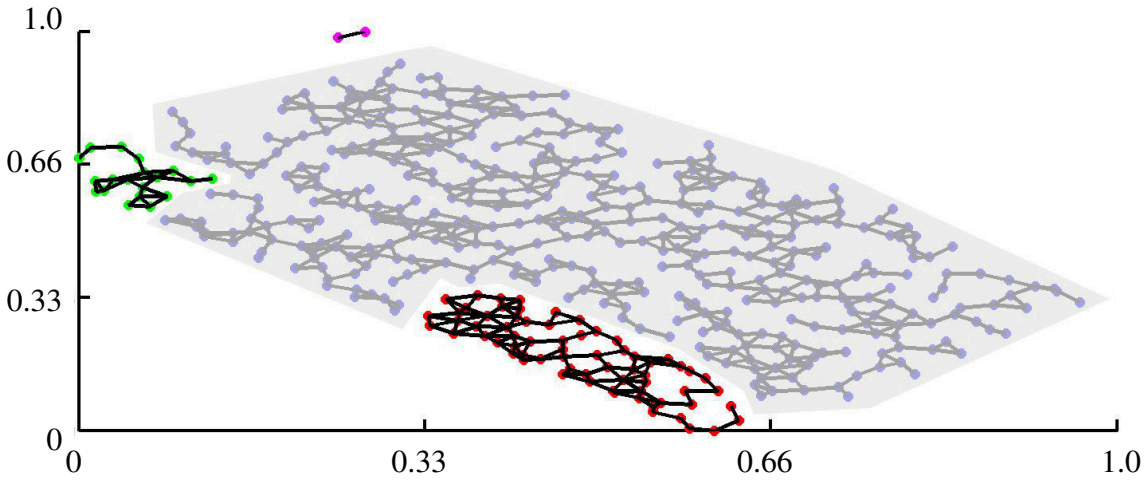


Fig. 5.7 Identification of natural subsets within the graph of the dataset. The grey area highlights the largest subset with smaller, peripheral sets to the north, west and south.

5.3.2 Benchmark testing

To establish a benchmark for the proposed algorithm, two existing graph traversing methods and a solution to the traveling salesman problem are implemented.

The proposed algorithm (abbreviated KPSPP) was compared against a Depth First Search (DFS) [60], Breadth First Search (BFS) [60] and a traveling salesman implementation (TSP-NN) [62] based on the nearest neighbor approach. Algorithms were compared by computing a route between randomly selected start and end points on the convex hull. Evaluations at two different scales were conducted; one investigating the small set performance and a second investigating large set performance. In the small set tests, algorithms were run on 10 datasets containing 100 to 1000 trees. In the large scale testing 14 datasets of differing size, containing between 3500 and 36 000 trees, were performed. In these tests, the datasets were measured data from 14 plantation forests, not simulated data. Results for the small scale testing are listed in Table 5.1 with summary statistics listed in Table 5.2. Similarly, for the large scale testing, results are summarized in Table 5.3 with summary statistics listed in Table 5.4.

It is expected from the testing that DFS and KPSPP have similar performance as KPSPP implements a planning approach similar to DFS at its core. As the target application of a BFS is traversing a graph in a short and fast manner it is expected that this will return paths covering much less nodes than the other approaches considered. TSP-NN, being an exhaustive search through the nodes, is expected to return the highest coverage paths but take

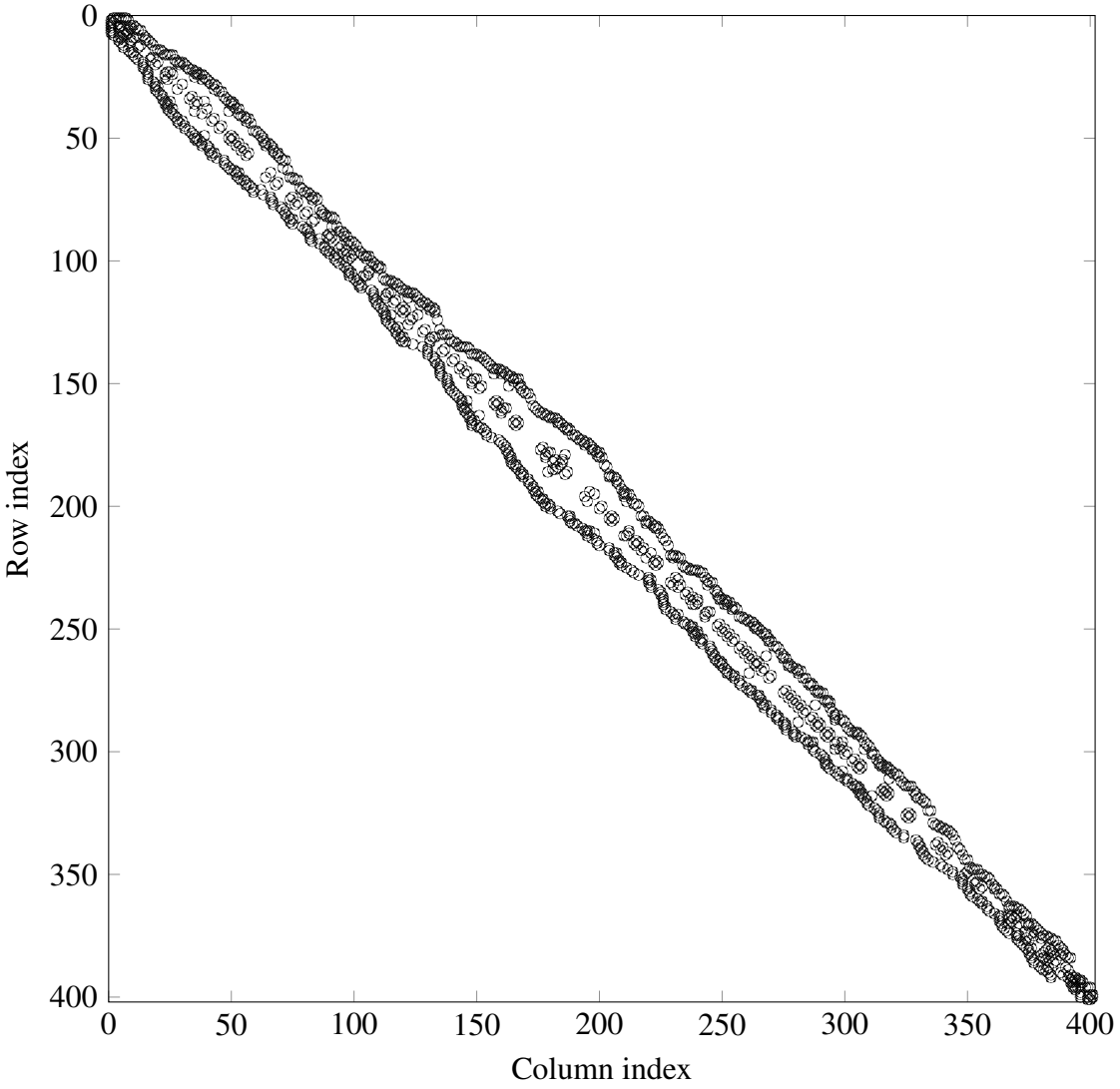


Fig. 5.8 Occupation of the adjacency matrix. Black indicates non-zero elements and white indicates zero-valued elements.

the most time to complete.

5.4 Discussion

The use of sparse matrices is justified by the low occupation of the adjacency matrix, observable in Fig. 5.8. As the matrix contains edge data for 401 nodes and in this case contains 1274 non-zero elements, 0.8 % of the available elements contain relevant edge data. In the standard matrix implementation, 99.2 % of the space would describe edge costs for non-existent links, they are zero valued. The figure also shows that the data is tightly distributed about the main diagonal indicating that edges between trees are connected to adjacent trees – as expected from the Delaunay Triangulation definition of edges. Due to the unidirectional edge assumption the matrix is symmetric along the main diagonal, this allows for the rapid formation of the entire matrix using one half to generate the other as $\boldsymbol{\alpha}_{(i,j)} = \boldsymbol{\alpha}_{(j,i)}$.

From the small scale testing it can be concluded that the DFS and KPSPP performed similarly well with respect to coverage and standard deviation. TSP-NN outperformed KPSPP on the basis of nodes visited (100 %) however required five times more operator interventions (6.86 compared to 1.36) and taking an order of magnitude longer per node. DFS required no interventions but as it covered approximately the same amount of nodes as KPSPP, outperformed KPSPP at this scale. The BFS, although having no interventions, returned paths too short to be considered for coverage planning. At the small scale, the algorithms were ranked as: DFS/KPSPP, TSP-NN then BFS (best to worst).

In the large scale testing the TSP-NN algorithm could not be tested due to the calculation duration becoming impractically long, in the order of three hours. This shows that the TSP-NN is not practical at this scale. At large scales the KPSPP algorithm outperformed all others by covering more nodes in a more reliable manner. KPSPP obtained a mean coverage of 84.43 % with a standard deviation of 4.16, 88 % less deviation than the DFS. Although the DFS could produce high coverage paths, it also produced very short paths resulting in unreliability and therefore its large deviation. Again, as for the small testing, the BFS proved to produce paths too short for coverage planning. At the small scale, the algorithms were ranked as: KPSPP, DFS, BFS then TSP-NN (best to worst).

Results show that the DFS algorithm can reliably find a route between the desired points over the range of set sizes. This algorithm returned paths which, on average, visited 56.65 %

Table 5.1 Small scale coverage results for each algorithm tested

		COVERAGE BY ALGORITHM, % (ERRORS IN ROUTE, %)			
SET	SET SIZE	KPSPP	DFS [60]	TSP-NN [62]	BFS [60]
1	100	73.00 (2.74)	35.00 (0.00)	100.00 (9.00)	3.00 (0.00)
2	200	55.00 (3.64)	67.50 (0.00)	100.00 (10.00)	7.50 (0.00)
3	300	76.00 (1.75)	59.33 (0.00)	100.00 (7.67)	5.33 (0.00)
4	400	14.00 (7.14)	37.25 (0.00)	100.00 (6.25)	2.25 (0.00)
5	500	73.40 (1.91)	10.80 (0.00)	100.00 (6.20)	3.00 (0.00)
6	600	36.83 (5.43)	16.00 (0.00)	100.00 (6.17)	2.00 (0.00)
7	700	16.29 (5.26)	19.86 (0.00)	100.00 (5.43)	1.43 (0.00)
8	800	11.38 (2.20)	22.75 (0.00)	100.00 (6.25)	0.63 (0.00)
9	900	20.67 (3.23)	76.33 (0.00)	100.00 (5.78)	1.78 (0.00)
10	1000	66.80 (3.14)	25.90 (0.00)	100.00 (5.90)	1.20 (0.00)

Table 5.2 Summary statistics for data presented in Table 5.1 on small scale testing

	KPSPP	DFS [60]	TSP-NN [62]	BFS [60]
MEAN COVERAGE (%)	44.34	37.07	100.00	2.81
STANDARD DEVIATION OF COVERAGE (PERCENTAGE POINTS)	27.28	22.92	0	2.10
MAXIMUM COVERAGE (%)	76.00	76.33	100.00	7.50
MINIMUM COVERAGE (%)	11.38	10.80	100.00	0.625
MEAN NUMBER OF ERRORS IN PATH (%)	1.36	0	6.86	0
MEAN TIME PER PATH ELEMENT (S)	3.2×10^{-2}	4.6×10^{-5}	1.1×10^{-1}	7.1×10^{-5}

Table 5.3 Large scale coverage results for each algorithm tested

		COVERAGE BY ALGORITHM, % (ERRORS IN ROUTE, %)		
SET	NODES	KPSPP	DFS [60]	BFS [60]
1	3500	83.26 (3.74)	65.60 (0.00)	0.71 (0.00)
2	6000	70.03 (3.88)	62.97 (0.00)	1.10 (0.00)
3	8500	83.42 (3.69)	53.02 (0.00)	0.53 (0.00)
4	11000	84.57 (3.92)	59.62 (0.00)	0.70 (0.00)
5	13500	86.16 (3.44)	81.64 (0.00)	0.59 (0.00)
6	16000	84.51 (3.52)	5.69 (0.00)	0.21 (0.00)
7	18500	86.46 (3.67)	58.35 (0.00)	0.21 (0.00)
8	21000	86.24 (3.59)	38.65 (0.00)	0.49 (0.00)
9	23500	85.03 (3.62)	61.62 (0.00)	0.13 (0.00)
10	26000	86.01 (3.76)	40.24 (0.00)	0.35 (0.00)
11	28500	87.25 (3.60)	55.12 (0.00)	0.22 (0.00)
12	31000	86.89 (3.70)	63.02 (0.00)	0.26 (0.00)
13	33500	86.07 (3.66)	63.61 (0.00)	0.51 (0.00)
14	36000	86.16 (3.59)	83.99 (0.00)	0.35 (0.00)

Table 5.4 Summary statistics for data presented in Table 5.3 on large scale testing

	KPSPP	DFS [60]	BFS [60]
MEAN COVERAGE (%)	84.43	56.65	0.45
STANDARD DEVIATION OF COVERAGE (PERCENTAGE POINTS)	4.16	18.56	0.26
MAXIMUM COVERAGE (%)	87.25	83.99	1.1
MINIMUM COVERAGE (%)	70.03	5.69	0.13
MEAN NUMBER OF ERRORS IN PATH (%)	2.92	0	0
MEAN TIME PER PATH ELEMENT (S)	3.8×10^{-2}	3×10^{-6}	3×10^{-6}

of the available nodes and achieved as high as 83.99 % coverage. Although the DFS was able to reliably return paths, it had large variability in the coverage which is indicated by the standard deviation of 18.56. A factor affecting the variation is the quantity of outcomes such as paths covering 5.69 % as well as 3 out of the 14 paths covering less than 50 % of the nodes. This variation, the highest of all algorithms considered, decreases its dependability in large scale route planning. As the coverage result is not reliably high, repetition is required to find a suitable path. The DFS does not consider the location of the end point relative to the start point; if the end node is found in a branch traversed early then a short path is returned. Since the KPSPP is considerate to the start and end point locations, a path is planned that explores as much of the graph as possible before arriving at the end point. Another disadvantage of DFS is that consideration is not given to the heuristically valued edge costs as specified in the adjacency matrix. For the DFS, the adjacency matrix is read in a binary fashion – true if there is an edge, false otherwise. The path cost is then the summation of edge costs once the path has been found, there is no on-line decision regarding the costs. Hence the algorithm does not make an optimization for the edge costs, it is single minded in seeking the end node.

The BFS always returned the shortest paths of all algorithms considered. At its maximum 1.1 % of trees were reached and reaching 0.13 % in the minimum case. For a coverage path planner, this was rejected as the goal is to maximize the number of nodes visited. The TSP-NN algorithm was found to consume impractical amounts of time for the larger sets of Table 5.3 (>3500 nodes), which ran in the order of hours. For small sets this algorithm achieved visitation to all nodes however did so by taking paths not within the machines reach. An average 6.86 % of the edges taken by the algorithm were outside of the workspace of the machine.

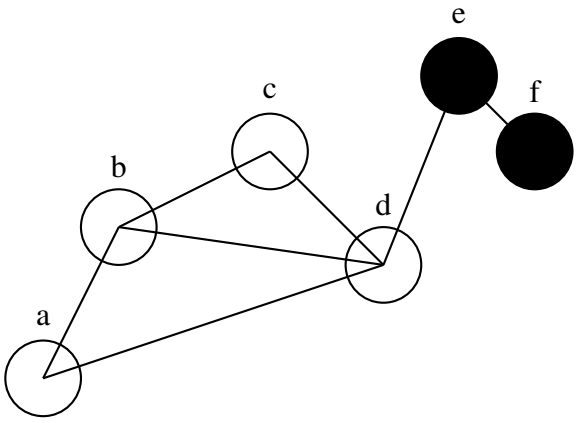
The proposed algorithm, KPSPP, returned paths covering >70 % of nodes for all datasets, covering 87.25 % in the best result. The mean coverage (84.43 %) and standard deviation (4.16) of this approach outperformed the next best algorithm, DFS, which obtained results of 56.65 % mean coverage and standard deviation of 18.56. With the proposed approach, a holistic view is taken by planning a path to visit as many subsets as possible before visiting the end point leading to better outcomes. Results for this method demonstrate that the number of nodes covered is much more predictable than other methods.

5.4.1 Limitations and future work

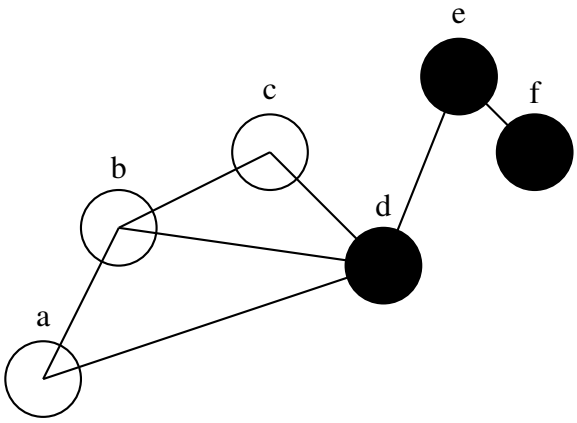
In the current implementation, subgroups are formed based upon their position. Position based grouping can lead to subgroups with poor intra- and/or inter-subgroup connectivity.

Poor connectivity, in turn, contributes to incomplete coverage paths due to limited accessibility. An alternative partitioning strategy is based upon connectivity, such that all nodes in a group are highly connected with each other. This could result in higher coverage paths.

Consider the two sets in Fig. 5.9 in which node d could be in either set. In case 1, node d is associated with the white set. In order to traverse from the white set to the black set, any path must do through node d. In the second case however, the black set can be entered by any of nodes a, b or c. The second case presents much more inter-connected subgroups with three connects rather than one. It is the second case the may present a stronger partitioning methodology.



(a) Case 1



(b) Case 2

Fig. 5.9 Two partitioning approaches for subsets

Due to the nature of the planning component of the algorithm, the maximum group size is limited. In practice, a maximum size of 20 nodes per planning event was found to be efficient.

Groups over 20 were found to increase the computation time and significantly increase the memory requirements.

The harvesting route was determined off-line, prior to the harvest occurring. Due to environmental factors the crop may change between the plan being created and executed. For example wind or rain might damage or remove certain parts of the crop. To account for changes in the crop in real time, an on-line implementation should be developed to work in tandem with the off-line plan. This would allow the off-line plan to evolve with changes in the field as they are discovered.

It is assumed that the operator intends to harvest all nodes. A complete harvest may not always be the intention due to factors such as ripeness or health. Currently there is no provision for including nodes that assist with mobility but are not to be harvested. Such a feature would increase the agility of machines such as that in the case study as well as produce a higher quality yield.

This algorithm, coupled with the machine in Chapter 3, introduces a paradigm shift for forest managers. Future work with managers would involve integrating considerations into the planning schemes of new forests. Considerations such as the layout of roading, extraction and strategic placement of traversal aids would be foremost. Traversal aids such as fencing posts in strategic locations could ensure that the machine can traverse between all natural subsets. The design of existing planting methodologies may also change to take advantage of the idiosyncrasies of the proposed system.

5.5 Conclusion

An algorithm, KPSPP, has been developed to generate routes for autonomous harvesting robots. The proposed algorithm has been shown to be effective for the planning of discretized crops in three dimensions which had not yet been available. A case study of an autonomous tree-to-tree felling machine is formed. The study tested in fourteen different plantation forests with up to 36 000 trees. In the tests, the KPSPP algorithm out-performed all other algorithms in larger sets, with mean coverage of 84.43 % for 3500 to 36 000 trees. With a tighter standard deviation (4.16 percentage points) than the next best competitor (18.56), the algorithm proved effective and robust for planning large sets. Performance of the proposed method at lower set sizes (100 to 1000 trees), was less competitive where other existing methods proved equally viable. Increasing performance in smaller sets and refining the

subset formation methodology remain as future work.

Chapter 6

Sensor guided brachiation for a tree felling robot on steep terrain

Operation in a commercial plantation forest requires robotic machines to be capable of identifying the position, size and orientation of trees. An approach is proposed to identify the aforementioned tree properties using data gathered exclusively from a Light Detection And Ranging (LiDAR) scanner. Trees are detected in the gathered point cloud by searching for horizontally directed normal vectors. Cylinder fitting is performed using random sample consensus and three metrics are used to evaluate cylinder suitability. This approach was verified using eight different forest scenes with widely varying terrain to test its effectiveness. RMS error in trunk models was 3.10 cm*, whilst RMS error in location estimates was* 0.428 m*. These results present a* 76 % *to* 89 % *improvement over other literature. The system is then applied to brachiation for a novel forestry robot in order to identify suitable trees.*

6.1 Introduction

In order for robotics to assist in commercial forestry environments, the means to detect the trees is fundamental. This contribution seeks to enhance the robustness of a global route determined from aerial LiDAR data a-priori as proposed in [107]. As the aerial data is not necessarily taken at the same time as the machine's mission, integrity of the route can not be guaranteed. The proposed route may include now damaged trees, false positives, false negatives and be inaccurate regarding the ground position of the tree. Terrestrial LiDAR, mounted onto a mobile robot, will be used to compliment the aerial data for re-planning based upon up-to-date information. Terrestrial data offers current data, well resolved with respect to the robot and will update the global map on trees are unsuitable or not found in the

aerial scan. The proposed approach is able to detect trees without knowledge of the ground plane and requires only one scan from a single location. Tree identification without express knowledge of the ground plane is an advantage in a forestry context as the ground is often obstructed with dead-fall trees and low lying shrubs. Using a single scan results in a faster and more efficient system as it does not need to maneuver about to capture point clouds from multiple viewpoints.

This contribution presents a novel tree detection methodology for diverse forest environments. An algorithm is developed to detect all trees and select only that which are suitable for the machine to grasp given its kinematic constraints. Upon field deployment, on-board terrestrial LiDAR is used to fully resolve a-priori calculated route and update on-line. This presents cost savings to the operator in terms of decreased cycle time between felling operations.

6.2 Methodology

In this section the methodology of gathering and analyzing the data, leading to environment detection, is presented.

6.2.1 System architecture

The system architecture (Fig. 6.1) to support semi-autonomous robotic operation may be described as a pipeline with four principle components: Terrestrial detection, route planning, motion planning and motion execution. This system is implemented in the Robotic Operating System (ROS) framework. The components are as follows:

Terrestrial detection: The detection node uses the LiDAR apparatus to scan the environment. The recognition algorithm, implemented with Point Cloud Library (PCL), returns objects identified in the point cloud as trees, ground or miscellaneous.

Virtual environment: LiDAR scan data is mapped back to the global frame by applying inverse kinematics. This is based upon the current current pose of the robot and the calibrated position of the LiDAR on the arm. Points in the LiDAR scan are then presented to the control system in absolute coordinates to the origin, not relative coordinates from the LiDAR scanner. Differences in pose of the scans are taken into account at this stage, thereby building a complete view of the surrounding environment.

Route planning: Routes throughout the forest must be determined at the global level to achieve the forest managers mission objectives. This is based upon aerial LiDAR data captured over the target area. Global planning [107] ensures that the machine traverses trees in an optimal order, avoiding dead-ends.

Motion planning: The 'Motion planning' node computes the trajectory of the machine between two trees suggested by the mission plan and confirmed by the operator. This trajectory is calculated using the state of the machine and estimated state of the environment such that it plans to avoid collisions. As the forest is a complex environment, the operator oversees this process.

Motion execution: The 'Motion execution' node performs tests on the motion request to ensure that it is within boundary conditions and manages the machine throughout motion using PID controllers for joints. If the machine requires fine local adjustment by the operator, this arbitrates access.

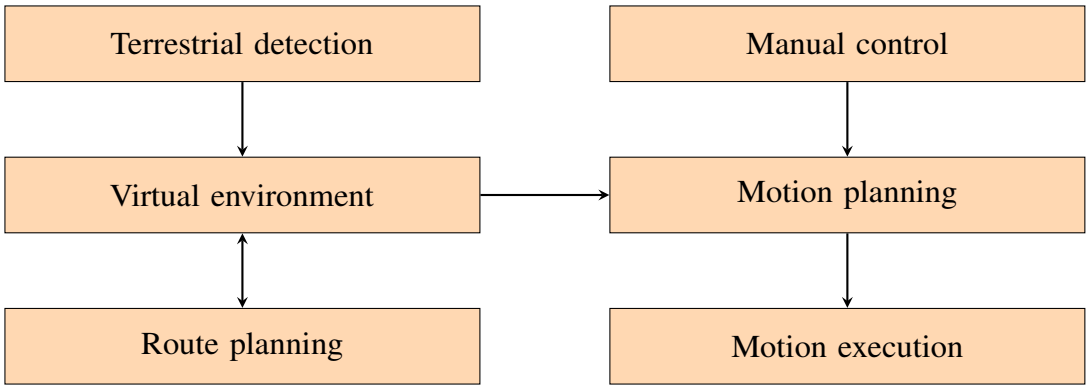


Fig. 6.1 System architecture for motion execution

6.2.2 Scanning apparatus

The scanning apparatus is shown in Fig. 6.2. A Hokuyo UTM-30LX LiDAR (Table 6.1) was used to gather depth information. Positioning the scanner in this manner minimizes the presence of dead-zone in the required field of view. The scanner has a 2-dimensional field of view; in order to generate a 3-D model, 2-D slices are accumulated during a 180° rotation of the servo-motor. The servo-motor (Dynamixel MX-28R, Table 6.2) encoder tracks the position. A high definition Logitech c920 camera provides real-time visual feedback to the operator. In order to protect the equipment during collisions, a pair of aluminum covers surround the sensors. An electronics enclosure is manufactured from 3-D printed ABS plastic to develop a sacrificial component that can be readily replaced should damage occur.

The servo-motor requires calibration in order to rotate the payload smoothly. The servo has four properties: "speed", "slope", "margin" and "punch" [108], see Fig. 6.4. The slope parameter defines the proportional gain, margin determines the error tolerance (or 'deadzone') and punch specifies the minimum torque applied once the margin has been exceeded.

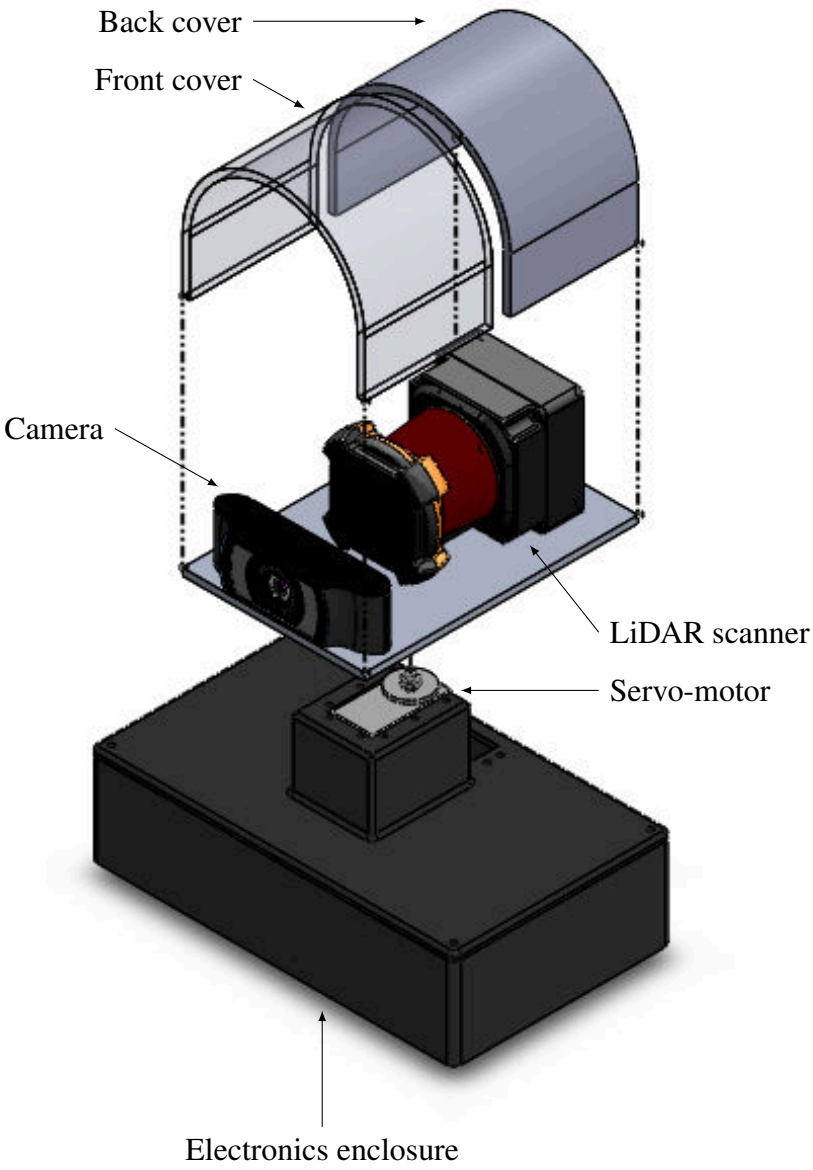


Fig. 6.2 Exploded view of the apparatus, adapted from [88]

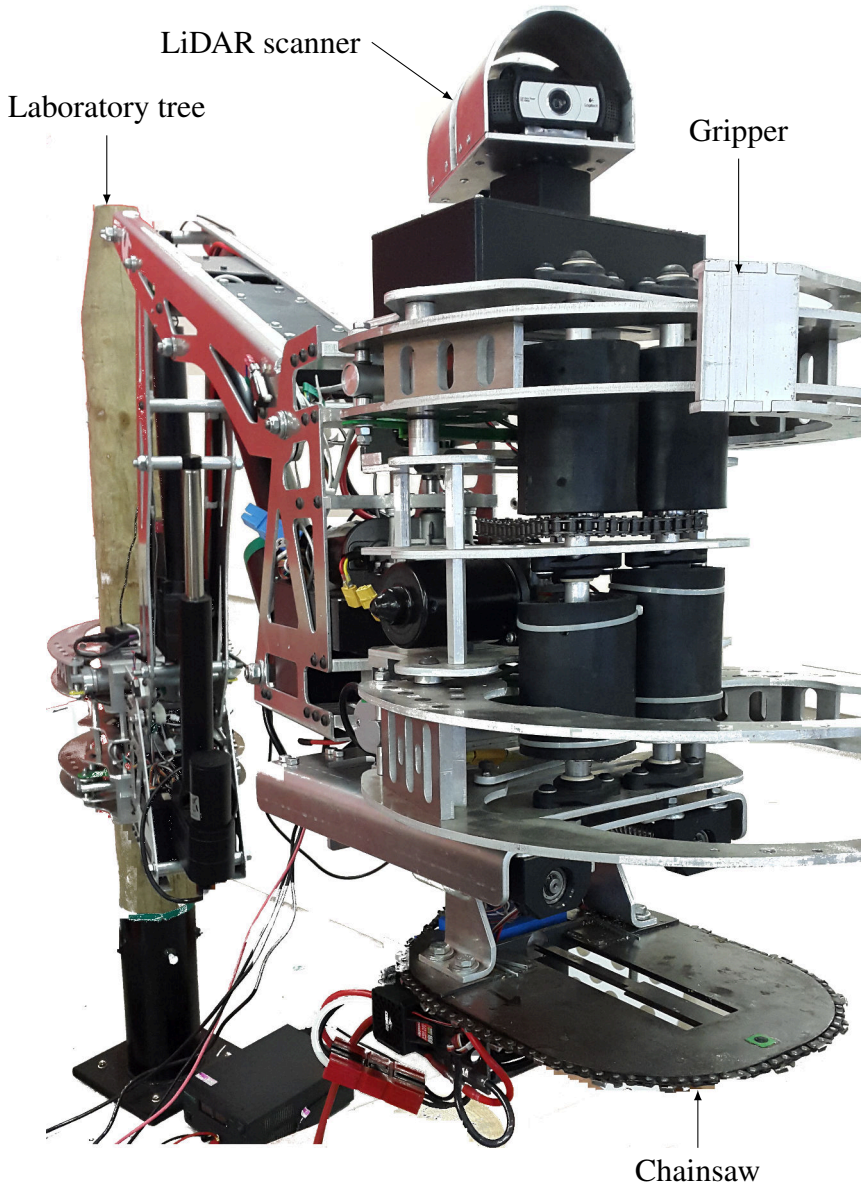


Fig. 6.3 LiDAR mounted on machine gripper

Table 6.1 Properties of the Hokuyo UTM-30LX LiDAR unit

Property	Value
Range (Minimum)	0.1 m
Range (Recommended maximum)	30 m
Range (Absolute maximum)	30 m
Measurement resolution	0.001 m
Angular resolution	0.25°
Angular field of view	270°
Scan period	0.025 s

Table 6.2 Properties of the Dynamixel MX-28R

Property	Value
Stall torque	2.5 N·m
Recommended maximum torque	0.5 N·m
No-load speed	55 RPM
Angular resolution	0.088°

The servo rotates with angular velocity ω, whilst the system captures 2-D slices during the movement. The distance d_p between two planes at a range r from the scanner can be found with their angle θ as in Eq. (6.1) and $\theta = \omega \times t$ where t is the period of the LiDAR scanner.

$$d_p = 2r \sin\left(\frac{\theta}{2}\right) \tag{6.1}$$

The speed at which the servo-motor should rotate the scanning platform is determined by the maximum allowable separation between scan planes and therein the scan resolution. Solving Eq. (6.2) provides the maximum speed at which the sensor platform should be rotated in order to achieve the desired scan resolution.

$$\omega \leq \frac{2}{t} \operatorname{asin}\left(\frac{d_p}{2r}\right) \tag{6.2}$$

6.2.3 Tree detection algorithm

The tree recognition pipeline (Fig. 6.6) contains four stages: filtering, characterizing, clustering and model fitting. An example LiDAR scan is shown in Fig. 6.7, this must be processed

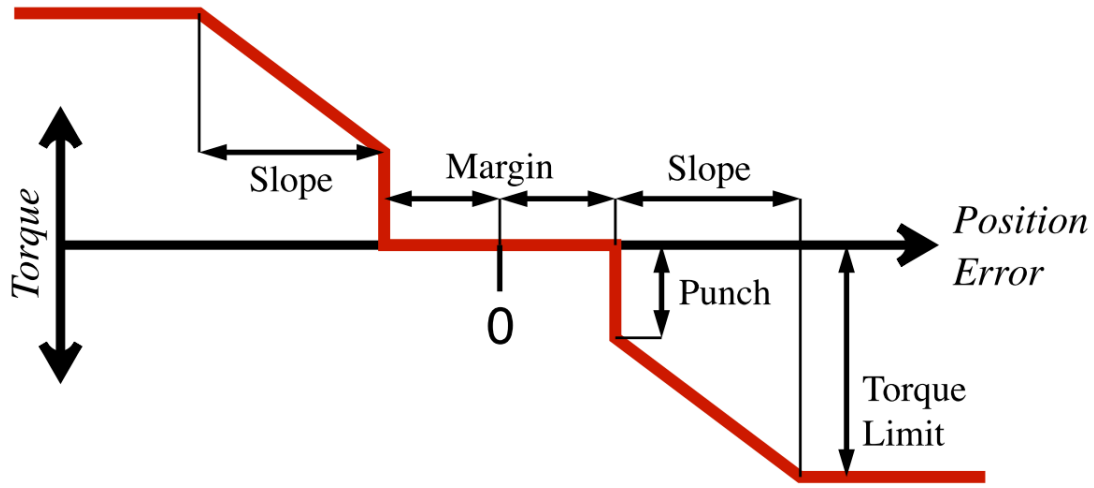


Fig. 6.4 Dynamixel parameters [108]

in order to identify the trees.

Filtering

Superfluous information is reduced by filtering and down-sampling. This is done in four ways; removing all points outside the range of interest (r_s), a voxel grid filter, outlier removal and ground plane identification. The desired scan resolution was determined, therefore the leaf size is set no larger than this value so as to maintain adequate resolution on tree trunks out to r_s. Outliers and small points of groups are generated by the weak returns of canopy and ground cover. Outliers are identified by having a mean distance from the investigated subset outside of 0.5 standard deviations from the group mean. Finally, removing the ground plane reduces the search space for trunks. A plane is fitted to the data using Random Sample Consensus (RANSAC) estimation. The ground is assumed linear with a thickness allowance for deviations and sampling errors.

Characterization

A unique feature that distinguishes trees from many other objects in the environment is their vertical alignment. A Pine tree will be principally comprised of vertical faces with horizontally aligned normal vectors. By considering this characteristic throughout the data, candidates for cylinder fitting may be identified and other points rejected. As a single point

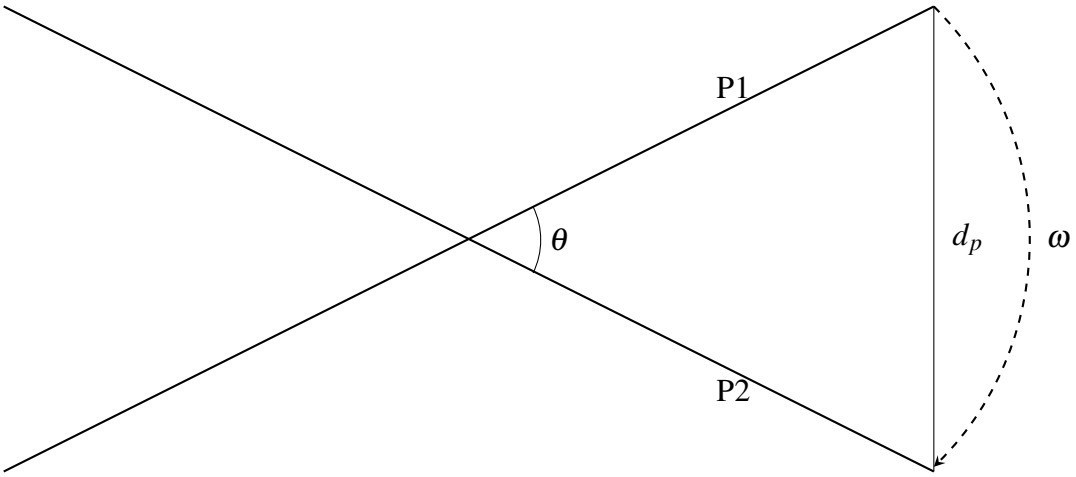


Fig. 6.5 Relationship between adjacent LiDAR scans

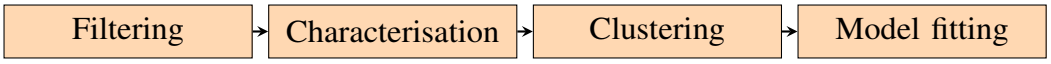


Fig. 6.6 Tree detection algorithm top-level flow chart

cannot mathematically have a normal vector, a local approximation is calculated. By forming a surface at that point consisting of neighboring points, the normal of that derived surface can be calculated. The range of the local approximation is determined by the target object size. The nearest k neighbours from a point are used. A k-d tree is used as "range queries can be answered in $O(\sqrt{n}+k)$ time, where n is the number of points in the k-d tree and k is the number of answers to the query" [109].

Clustering

Computational complexity of the subsequent model fitting step is reduced by identifying point clusters as candidate bodies. To identify clusters, regions are grown from random seed locations. Regions are assumed to have approximately collinear or slowly varying normals e.g trunks and walls. Therefore regions are grown with adjacent points with consistent normal vectors. Tree trunks are expected to have gently changing normals, whilst leaf clutter would have diversity in vector orientation. This results in regions with varying sizes. For a robust model fit, the clusters are constrained to a minimum size, below which data is insufficient to produce a model. Likewise, clusters that exceed a maximum size are likely to have been misidentified - for example a dense canopy and can likewise be excluded as trunks.

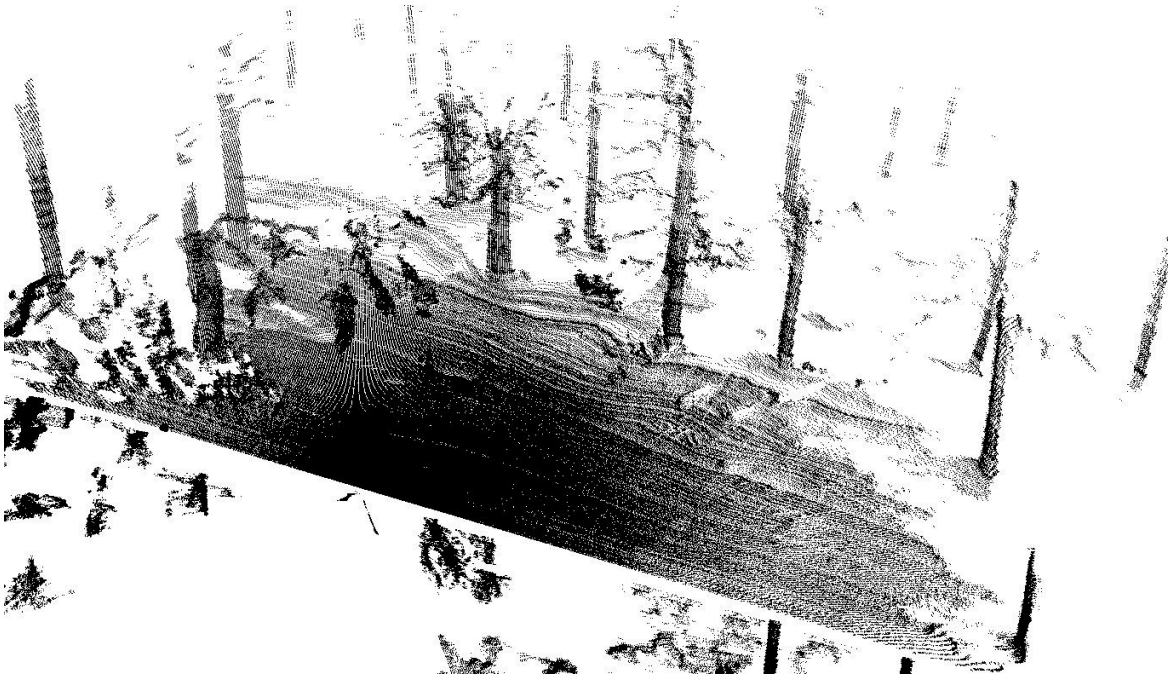


Fig. 6.7 LiDAR scan of a forests scene

Model fitting

For each of the clusters identified, cylinder fitting is attempted. An example cloud cluster is shown in Fig. 6.8. A random sample consensus is used. The fitting proceeds to modify the cylinder axis and radius until a consensus is formed from the population which minimizes the error between cylinder surface and points.

The proposed cylinder goodness of fit is interrogated by three classifiers. Firstly, if fewer than $\xi\%$ of the points in the cluster were used to model the cylinder, it is deemed to be of poor fit and rejected. Secondly, a threshold on the tree's orientation relative to gravity θ_v is used to exclude models that are not compatible with the machines grasp kinematics. Finally, the radius of the fitted cylinder must be grasped by the machine and so $r_{t_{min}} \leq r \leq r_{t_{max}}$. The first classifier considers point cloud properties whilst the latter two consider the limitations of the machine kinematics.

6.2.4 Integrating with planning systems

The machine requires a global plan in order to navigate the forest in a commercially effective manner. Global route planning maintains the desired future route throughout the forest in order to reach the goal position whilst achieving the mission objectives of the forest manager. An a-priori mission plan is generated based upon estimated tree positions. Estimates are

Fig. 6.8 LiDAR points incident with a tree trunk

based upon tree apexes detected in aerial LiDAR data, then extrapolating these down to ground level. Using the technique described in [107], the route throughout the forest can be optimized. As trees are detected by the terrestrial LiDAR, their previously estimated position (from aerial LiDAR) can be refined. The global route is recalculated on-line in real-time to maintain optimal traversal.

6.3 Results and Analysis

In order to evaluate the performance of the methodology mentioned above, a random selection of eight scenes were tested and presented. Optical images are shown in Fig. 6.9 with summary statistics listed in Table 6.3.

6.3.1 Calibration

Servo-motor parameters: For the forestry applications it is required to be able to detect trees within 10 m (r_s) of the machine. At r_s, the desired minimum number of planes coincident with a tree is set to three. Given that the machine can grasp tree diameters up to 0.22 m, the distance between adjacent LiDAR planes is $d_p = \frac{0.22}{3}$ m. Solving Eq. (6.2) results in a maximum servo-motor speed of $\omega = 0.29\,\mathrm{rad\,s^{-1}}$. In order to oversample, $\omega = 0.2$ was

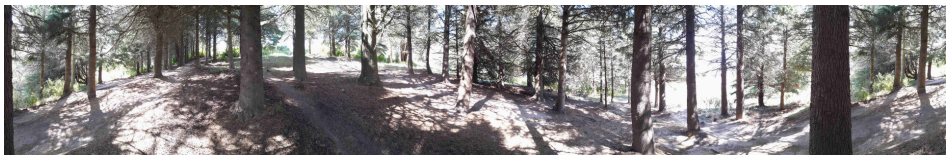
(a) Scene 1

(b) Scene 5

(c) Scene 2

(d) Scene 6

(e) Scene 3



(f) Scene 7

(g) Scene 4

(h) Scene 8

Fig. 6.9 Panoramic images of test scenes, taken at the scan origin

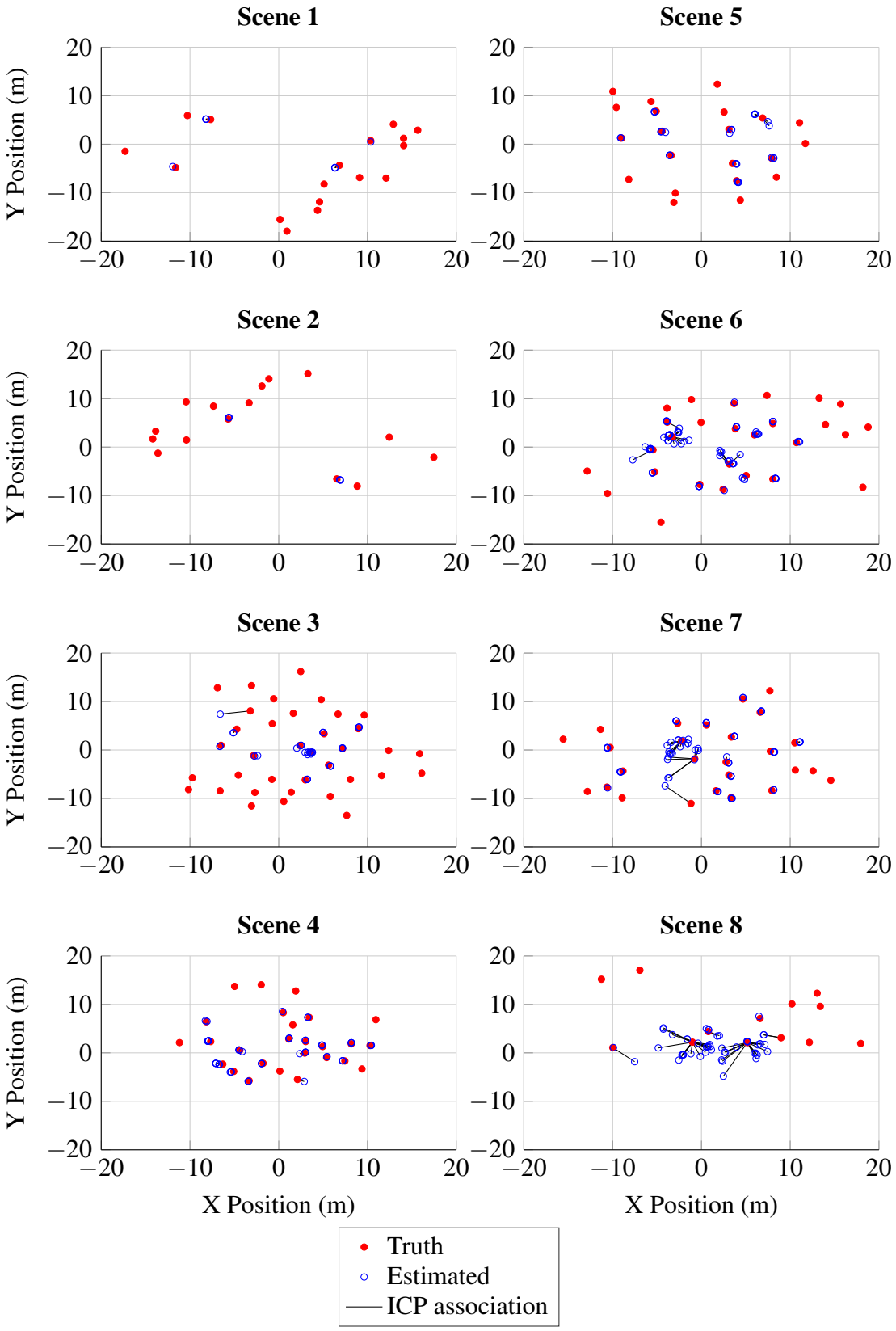


Fig. 6.10 Actual and modelled tree locations with closest tree assigned using ICP

Table 6.3 Summary statistics for each scene

Scene	Gradient (%)	Position RMS error, m	Trunk RMS error (cm)	PPV	Trees	Cluster quantity	Median cluster size	Filter reduction (%)	Duration (s)
1	10	0.567	2.74	1.00	17	14	579	50.0	0.91
2	6	0.370	2.82	1.00	15	14	621	49.1	0.91
3	3	0.349	3.16	1.00	36	20	710	44.6	1.14
4	9	0.337	2.39	1.00	26	36	563	41.1	1.58
5	6	0.352	3.22	1.00	21	40	623	39.9	1.85
6	5	0.473	3.41	0.90	27	59	588	42.4	2.61
7	14	0.411	2.91	0.93	26	75	523	40.8	3.42
8	11	0.504	3.93	0.59	13	48	645	46.8	3.05

used. In Fig. 6.11 shows the vertical view of a scan in which scan plane separation can be observed due to the linear, radial patterns throughout the data. The figure also illustrates the issue of shadows in the scan caused by foreground objects.

The error margin was set to the lowest level in order to maintain a high tracking accuracy. As the sensor platform presented a load much smaller than the servo capabilities, a low torque was required for the punch. The proportional gain was set at $50\,\mathrm{N\,m\,rad^{-1}}$ to produce smooth motion whilst scanning.

Algorithm parameters: The voxel size d_v must be less than d_p identified previously. To retain higher resolution away from the scanner, d_v was selected to be 0.03 m.

6.3.2 Tree Trunk Detection Algorithm Performance

The scanning radius r_s was set to 30 m in order to evaluate the performance at the maximum range of the scanner. The result shown in Fig. 6.12 illustrates trunk models mapped back onto a point cloud of a scene.

Data filtering: The initial filtering stages reduced cloud population by 39.9 % to 50.0 %. This consisted of the voxel filter, statistical outlier filter and ground plane removal. Tight coupling is shown between computational time to the input cloud size, with correlation coefficient of 0.9956. Weaker coupling exists to the number of clusters detected as indicated by a correlation coefficient of 0.7332.

Candidate clustering: Regions were grown to identify clusters in the cloud. Clusters containing less than 150 points were excluded due to insufficient information to make an informed decision. Regions stop growing when the change in normal is greater than 0.3 rad.

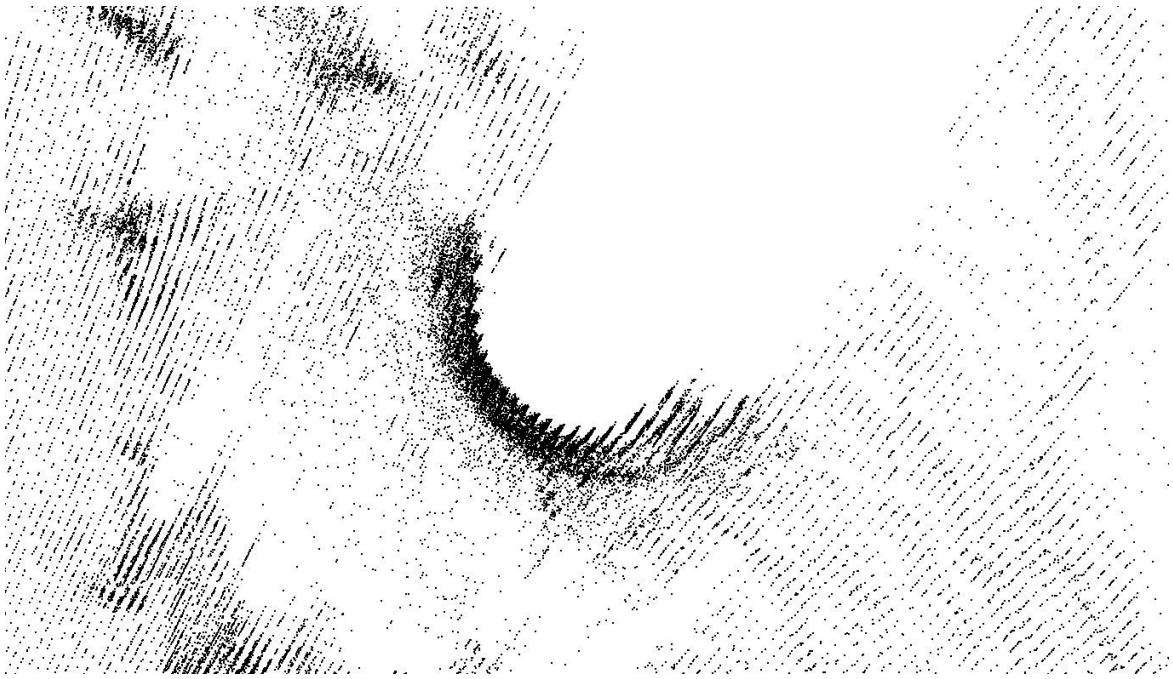


Fig. 6.11 Top-down view of LiDAR points incident with tree trunk showing shadow cast by trunk

Fig. 6.12 Example of tree models fitted to data

In scenes with high levels of canopy and/or ground cover, more clusters are formed due to the scattering of normals. The highest quantity of clusters was found in scene 7 due to the dense canopy, whilst in Scene 1 & 2 the minimum quantity was found due to the sparse tree distribution. Throughout all scenes, median cluster size differed between 523 and 710, with a median of 605 points. Within the model fitting process, no fewer than 77.5 % of points in the cluster were used, indicating strong performance of clusters isolating tree trunks including few foliage points.

Tree trunk model fitting: Cylinder models were proposed for each of the clusters found above. Suitability was determined using the three metrics of radius, vertical orientation and cloud utilization. The rate of each rejection per metric is shown in Fig. 6.13. The high frequency (64.8 %) of rejections due to the orientation classifier ($\theta_v \leq 15\,\text{deg}$) shows that a significant proportion of cylinders are not directly within the kinematic tolerances. This is due to input clusters of branches, clutter being fitted with cylinders as well as irregularly shaped trunks.

Models are also evaluated by their radius, to ensure the robot can grasp them. The data in Fig. 6.14 shows the model applied. Using this classifier, 9.0 % of the candidates were rejected.

Only 1.1 % of candidates were rejected due to insufficient utilization ($\xi > 35\%$) of data points for the model. This shows that the input clusters are being well modeled using the RANSAC cylinder.

The resultant RMS error between the stem model and point cloud (Fig. 6.15) was found to be 3.10 cm. The extrema for this error were found to be 0.73 cm and 5.52 cm. These results present a accuracy improvement over other work such as McDaniel et al [74] by 76 % to 89 %, in which accuracy of 28.9 cm, or 13.1 cm for $r_s < 13\,\text{m}$ was obtained by the authors.

Stem location estimation: The trunk positions for each scene were estimated as in Fig. 6.10. The iterative closest point procedure was used to assign estimates to truth data. Assignments with length greater than 1 m were rejected. The resultant RMS error in estimating the stem location was 0.428 m, with a minimum of 0.017 across all scenes. Error decreased in scenes 2-5 as the tree density increased near the sensor. Scenes 6-8 had notably higher error due to the much higher gradients and level of ground & canopy clutter. Scene 8 is a notable outlier in terms of stem position error as well as positive predictive value and trunk error.

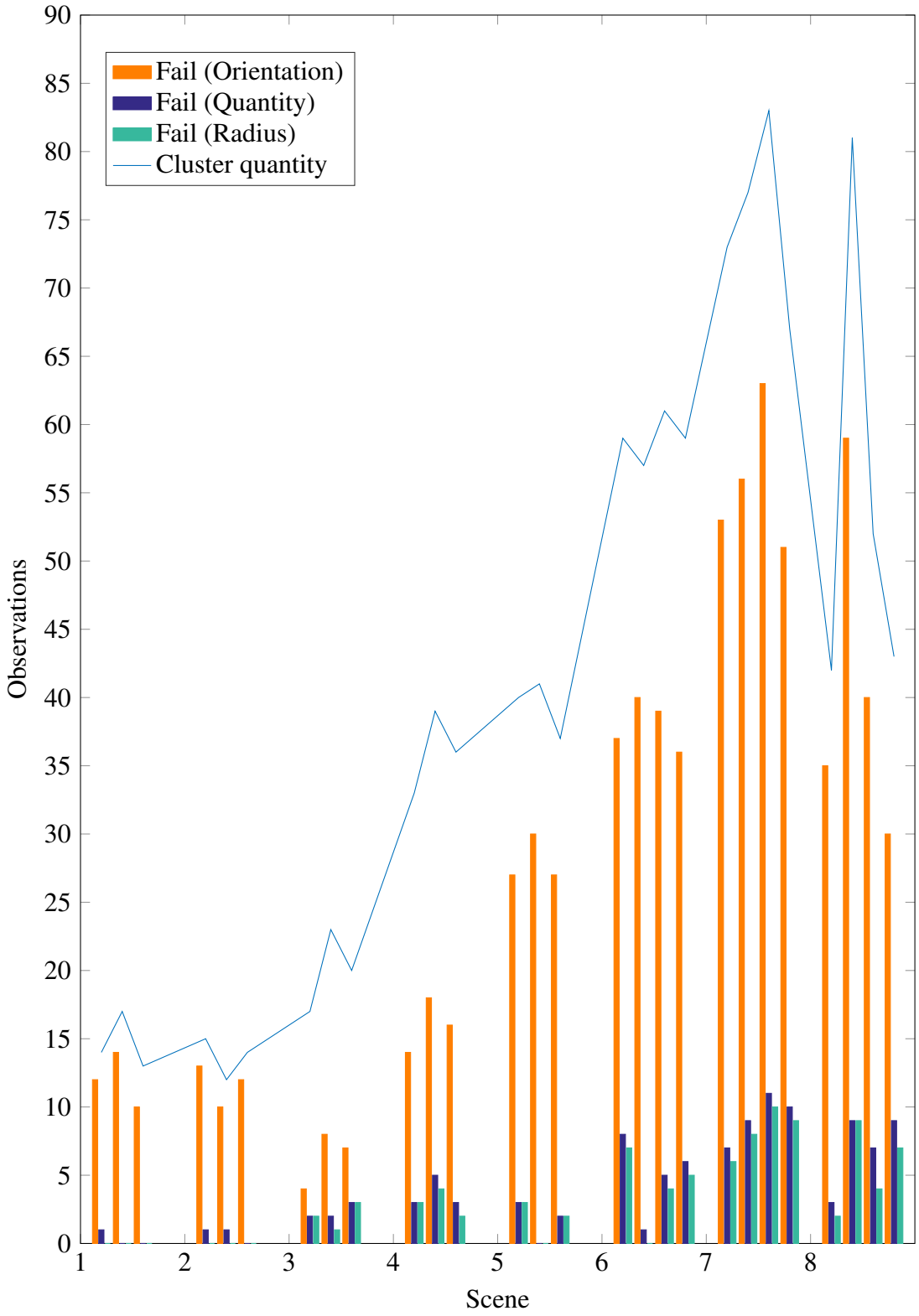


Fig. 6.13 Proportions of failure modes and cluster size. Repeated scan data shown for each scene.

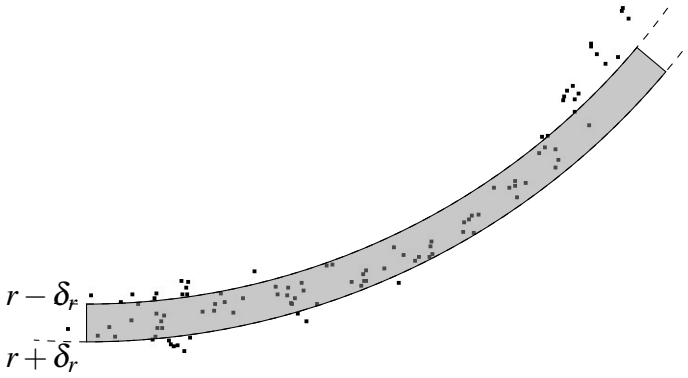


Fig. 6.14 Fitted points to cylinder model on 5 mm slice of data (Top view). Greyed area indicates region in which points are considered model inliers.

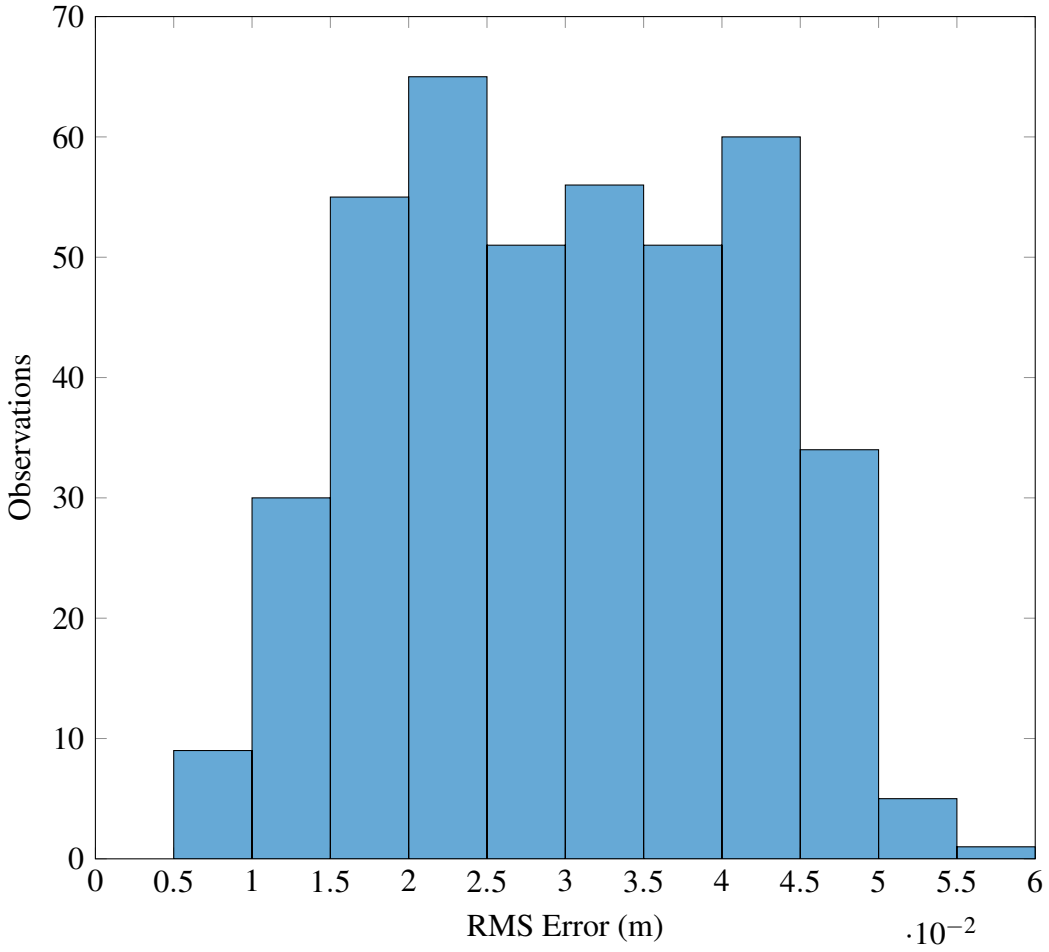


Fig. 6.15 Distribution of RMS error between tree model and data

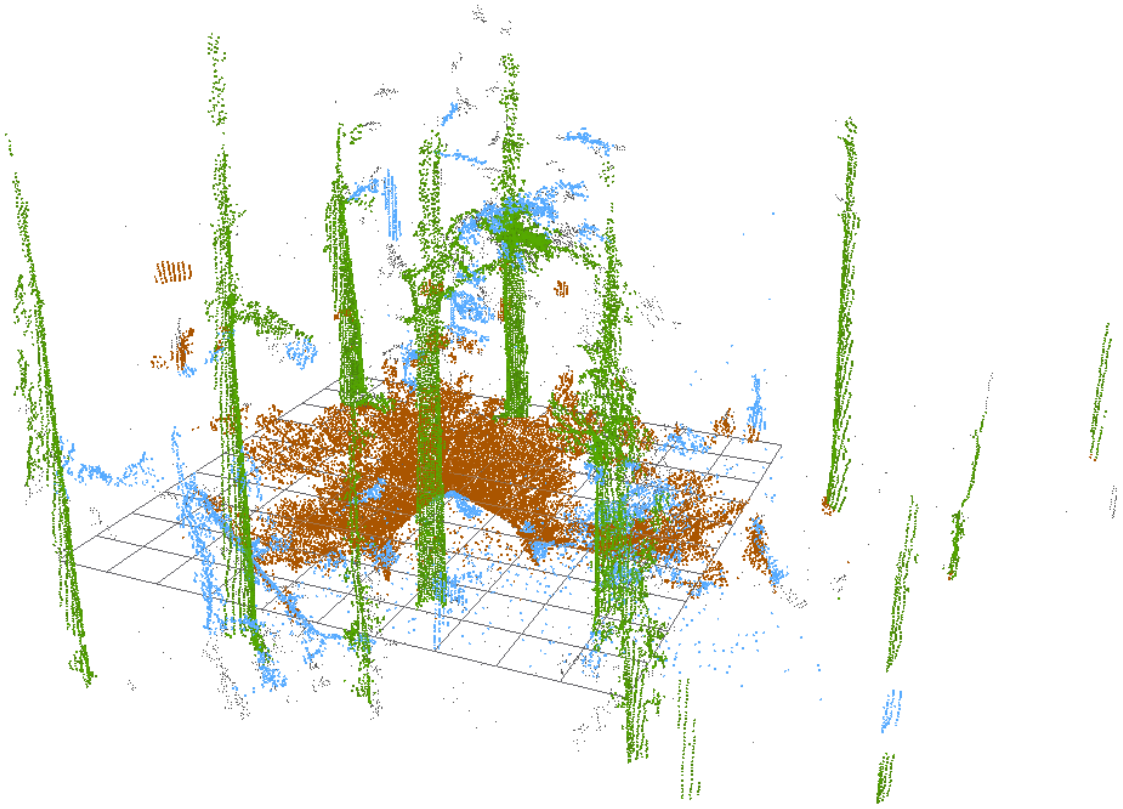


Fig. 6.16 Classified scene showing tree points (green), ground points (brown) and other points (blue)

6.3.3 Planning

Terrestrial data updates the aerial data through the operation. The target cycle time for the robotic platform to fell trees is 240 s. In the worst case, the machine recomputes its route after every fell, therefore the requirement for real time can be set as the target cycle time. A forest was scanned using aerial LiDAR which determined the tree locations as in Fig. 6.17. The section shown contains 67 trees. The route from the start (square) to the end (diamond) required 0.23 s to plan and covered $\frac{64}{67}$, (95.5 %) of the available trees.

Results for the model fitting are using in the trajectory planning for the robot. A trajectory generated using OMPL upon the LiDAR data is shown in Fig. 6.18. The route is planned from the robot's 'home' position to the position of a detected tree at (0.4909,1.3539,−0.8804).

The time taken for the scanner to capture the data is 15 s. Processing time for the cloud is up to 4 s. The remaining time in which the path must be recalculated is 221 s. It was shown in [107] that the planning requires an average 3.8×10^{-2} s per tree. Therefore the system can operate in real-time for a forest stand containing up to 5815 trees.

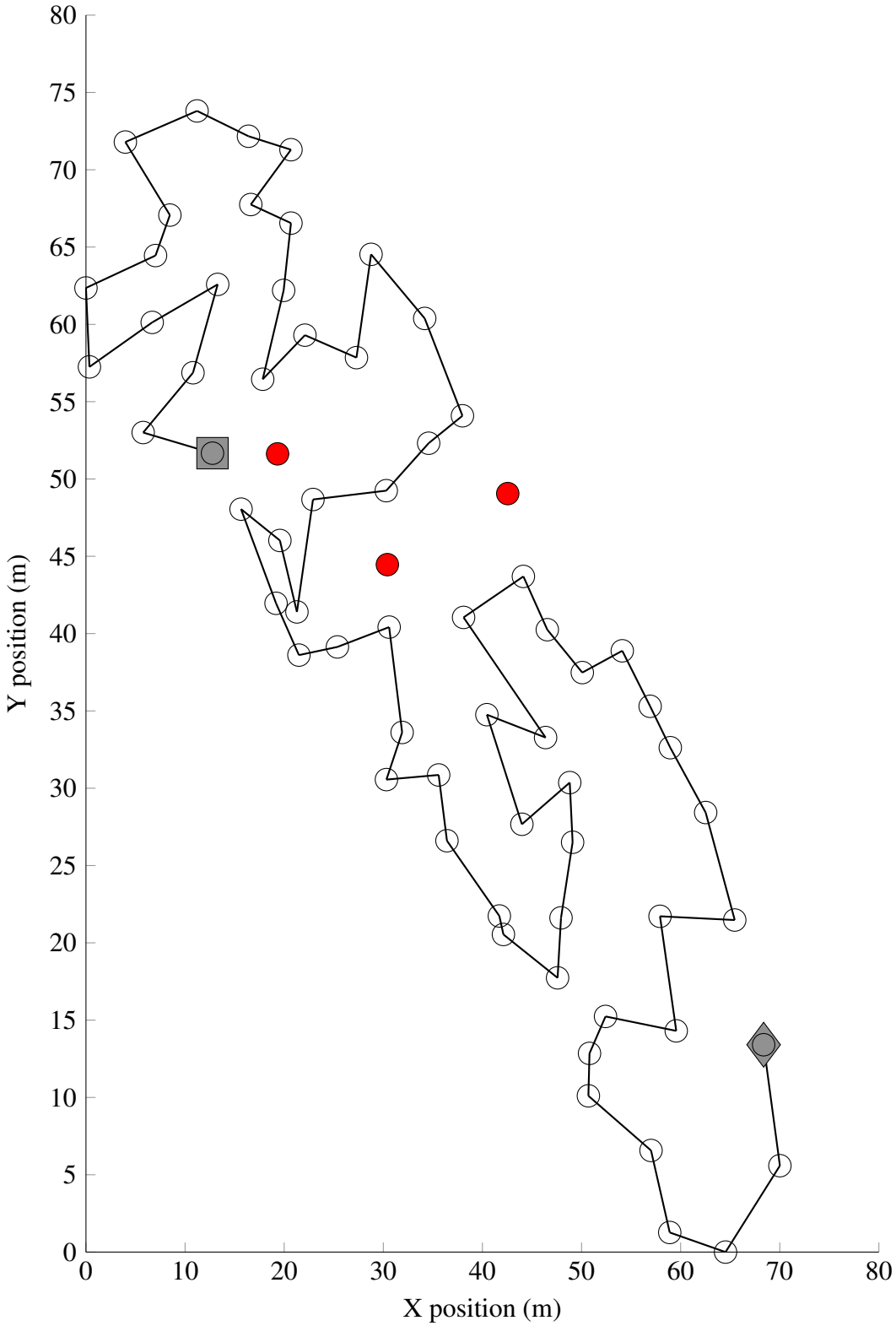


Fig. 6.17 Planned path for a forest block containing 67 trees. Square is initial tree, triangle is the targeted end-point of the route. Red-filled points indicate tree not traversed.

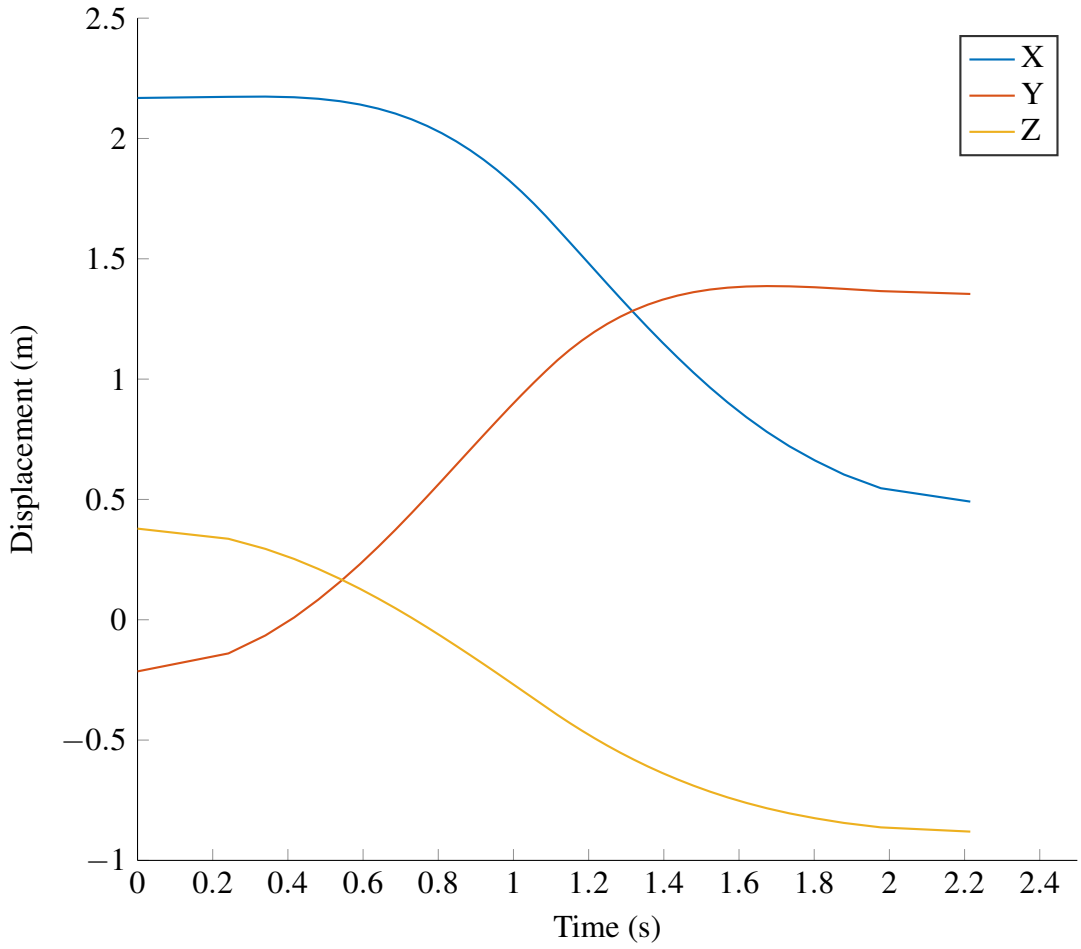


Fig. 6.18 OMPL planned end-effector trajectory

6.4 Conclusion

An algorithm has been developed to recognize Pine trees in plantation forest for a semi-autonomous robot. The identified trees have been selected from others such that they fall within the constraints of the machine for felling purposes. The system requires only a single scan, taken automatically, from a terrestrial LiDAR unit mounted on-board the robot. The algorithm takes advantage of the natural tendency of trees to grow parallel with the gravity vector by searching for faces that have horizontal normal vectors.

Compared to other results in the literature the algorithm is able to identify trees in diverse, cluttered environments using less scans than others such as that in [72]. Based upon this single scan the algorithm was able to process the largest point cloud (Scene 8), consisting of 801 927 points and 13 trees, in 3.05 s, whilst [70] processed a cloud of 301 287 points

containing 5 trees in 2 min. The algorithm achieved RMS error of 3.10 cm. Compared to other algorithms such as [74], the error is improved by 76 % to 89 %.

The system provides feedback to a-priori maps and route estimates gather from aerial LiDAR. Updating with terrestrial LiDAR measurements allows route re-planning in order to maintain optimal coverage important for harvesting. Routes were calculated in real-time for forest sections containing up to 5815 trees.

Future work will involve the fusion of optical camera data with the aim to increase quality in point characterization. If points can be more accurately identified as being trunk, ground, forest clutter or other then the rate of false positives and negatives can be improved. Better position estimation could be gained through optical edge detection as well as using piecewise cylinder models along the trunk so as to estimate it's ground position more accurately.

Chapter 7

Autonomous force-sensitive cutting mechanism for a tree felling robot

A new chainsaw mechanism has been developed for the recently published innovative tree-traversing robot [88]. A technique and corresponding control scheme for autonomous felling is proposed. A suitable cutting technique for this mechanism is derived by investigation of existing felling techniques. In order to position the chainsaw bar during felling, a full-state feedback controller is developed. An additional controller is designed to implement the felling procedure, which makes use of a force sensor integrated into the blade to detect tree lean. Tests conducted in a laboratory environment are used to evaluate the system, showing that this system is suitable for felling trees up to $0.22\,m$ *in diameter and achieves a position accuracy of* $0.2\,\text{mm}$*. Such a novel felling mechanism system, which has until now been unavailable, contributes significantly to the development of a lightweight tree harvesting robotic system.*

7.1 Introduction

In this contribution the design, control and implementation of a novel felling mechanism for robotic tree felling is presented. Unlike previous robotic chainsaw devices, this system can fell trees rather than cut branches. The chainsaw blade features a force sensor that is used to detect the downward tree force onto the blade, used for adjusting the control and monitoring the tree's state. Firstly, current industry techniques for felling are investigated in Section 7.2. This allows the derivation of an appropriate felling strategy for this application (Section 7.2.3). This derivation of a felling strategy is new as such a platform has until now been unavailable. The technique then motivates the chainsaw design in Section 7.3. A model

of the chainsaw mechanism is then derived in Section 7.4.1 allowing for a position controller to be designed. Subsequently the felling control will be implemented in Section 7.5. Finally, the test results are documented in Section 7.6.

7.2 Derivation of felling procedure

Established cutting techniques of motor-manually and mechanized felling will be examined in Sections 7.2.1 and 7.2.2 and for the tree-felling robot, a practical approach will be derived in Section 7.2.3.

Regarding the felling of a tree the terms front-cut, back-cut and kick-back shall be explained (Fig. 7.1): A front-cut is on the side of the intended direction of fall. A back-cut is on the opposite side to the front-cut. Kick-back describes the undesired case where the falling tree slides over its stump and the bottom of the trunk moves in the opposite direction as the tree falls. When felling is executed motor-manually, the tree butt travels uncontrollably towards the operator and may cause serious harm or even death to the logger [20].

The equations of motion for falling trees have been derived by Lyons, Sessions and Wimer [111]. Their work investigated the Humboldt and conventional cuts. The holding wood was found not to completely break until the undercut closes. At the point of the undercut closing, was the most dangerous time for kickbacks. Their models found that the Humboldt cut allows the tree to move sooner after the holding wood breaks. Papesch, Moore and Hawke [92] investigated the bending moments in *Pinus radiata*. Positive relationships were found between the resistive bending moment and the tree's height, diameter and stem volume. These factors will effect when in the back-cut the tree will fall.

7.2.1 Motor-manual felling review

There are many techniques for logging with hand-held chainsaws [111]. The choice of technique depends upon the size, natural direction of fall and surroundings of the tree, in [112] Hale provides a detailed review of cutting methodology. In the following, only techniques that are applicable for "generalized" trees are described. By generalized, it is meant the tree has no more than a slight lean after inclusion of all factors, it is healthy, not hung up or otherwise negatively influenced, such that a special cutting procedure would be required. Sophisticated felling techniques for these special cases are excluded in advance.

A standard motor-manual felling procedure consists of a front and a back-cut, Fig. 7.2 shows this procedure. The front-cut is done first to a depth of $25-33\,\%$ of the tree diameter and creates a so called kerf face [113]. The back-cut is always set above the FC and of such

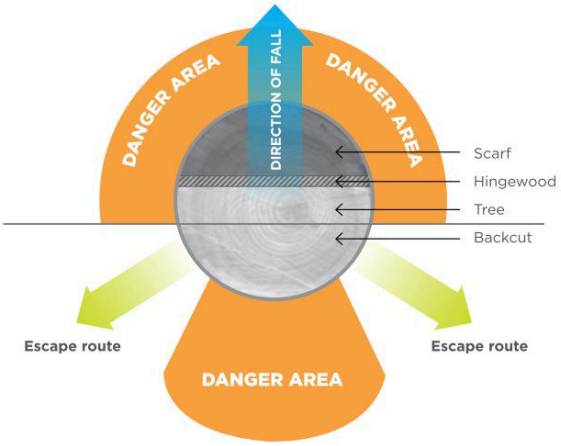


Fig. 7.1 Danger areas associated with tree felling [110]

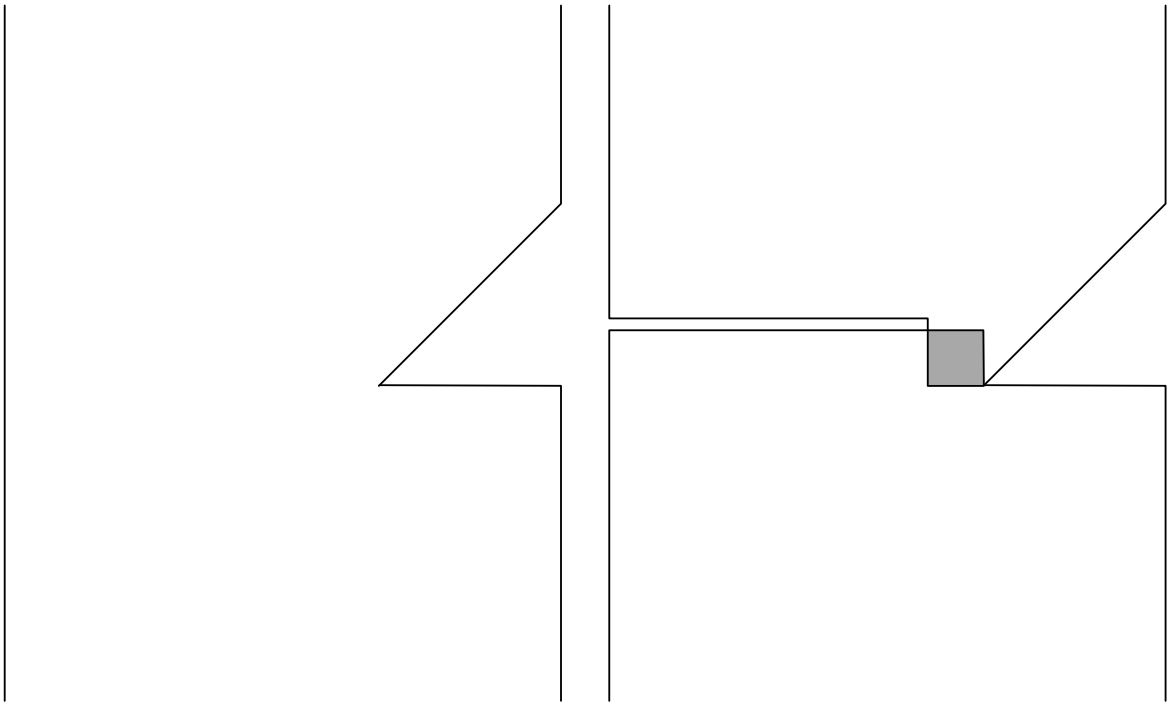
depth that approximately $1/10^{th}$, but at least 50 mm, of the wood will remain, the so called hinge wood. The tree will pivot on the hinge wood when falling [1]. Thereby the hinge wood is used as a guide for the direction of fall. When the tree falls, the kerf face closes which forces the hinge wood to break. From there the fall is uncontrolled. An advantage of the hinge wood is that it reduces the chance of kickback, this benefit increases with the Humboldt and open face method Fig. 7.3.

Different kerf faces have been established like shown in Fig. 7.3. Different front-cuts affect the falling behavior, variations of the back-cut are mostly due to the possible combinations of chainsaw size and tree diameter.

7.2.2 Mechanised felling review

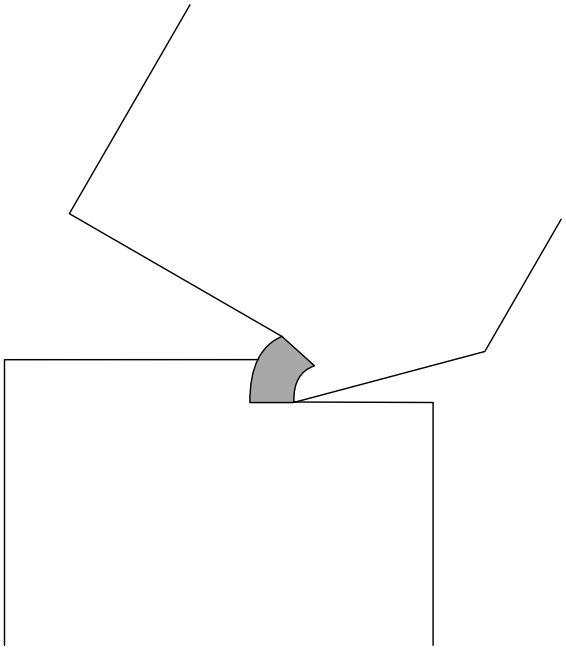
The approach of mechanized logging is different from motor-manual cutting methodology. Mechanized cutting styles are simpler as the machine controls the tree down to the ground. The front-cut is a simple, flat cut like the back-cut due to the inability of the felling heads to cut diagonally into a tree to create a kerf face.

Felling heads execute cuts by rotating a chainsaw sideways into the tree and there exist three styles (see Fig. 7.5) [20]. With small to medium size trees no front-cut is required, these trees are felled with one cut going completely through. For larger trees a front and a back-cut is required, which is called double cut. Last is the triple cut which is used for very large trees where a double cut does not sever the tree completely, the back-cut is split into two cuts.

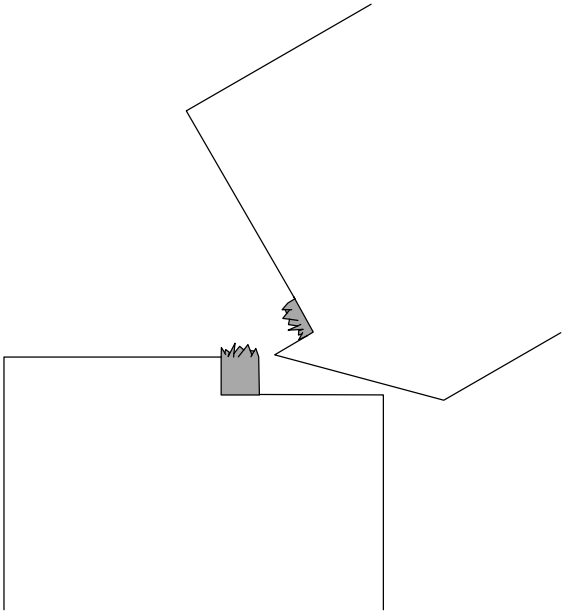


(a) Front cut (conventional face)

(b) Back cut, leaving the hinge wood (grey)

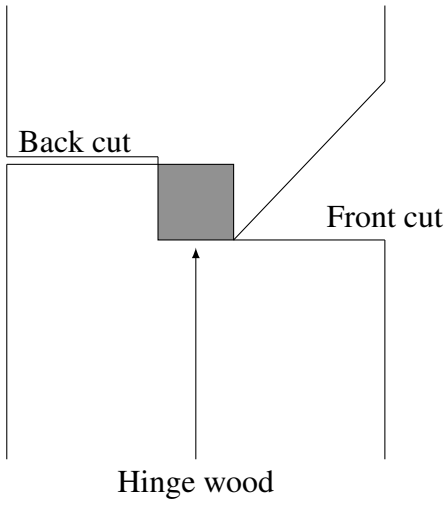


(c) Fall of tree guided by the hinge wood

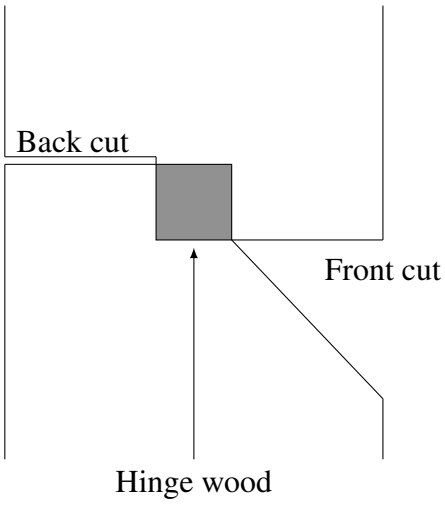


(d) Hinge wood breaks and the rest of the fall is uncontrolled

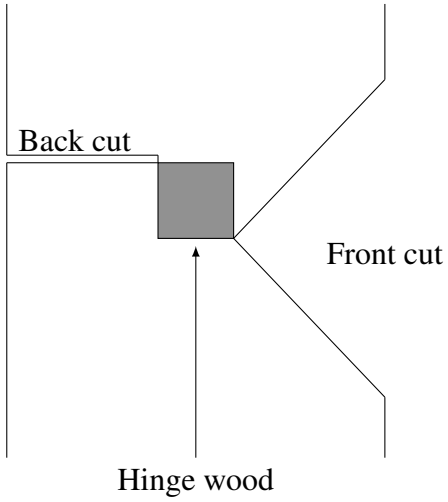
Fig. 7.2 The felling process in four steps



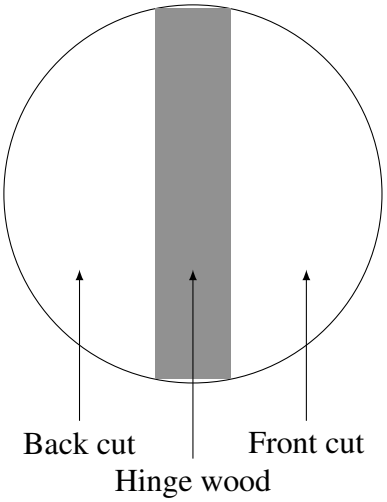
(a) Conventional face



(b) Humboldt face



(c) Vee face



(d) Top view on a trunk, which is identical for all kerf faces

Fig. 7.3 A selection of face types

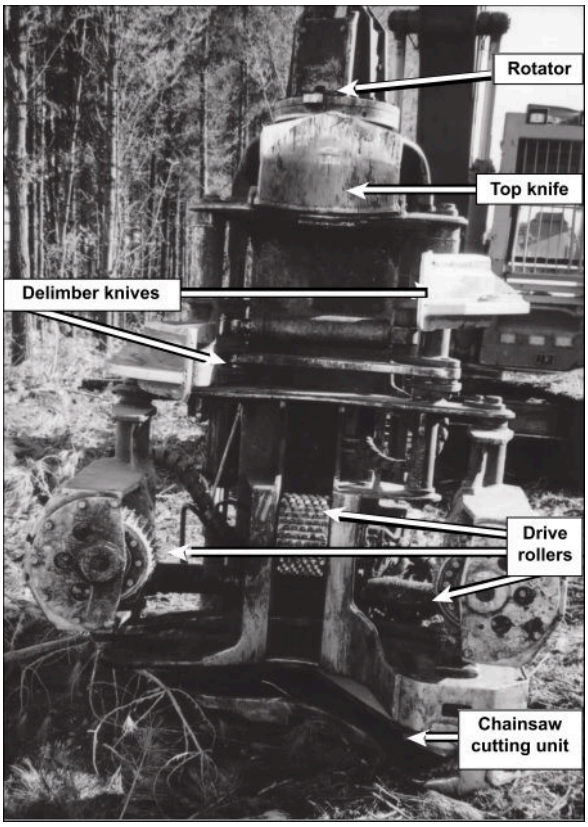
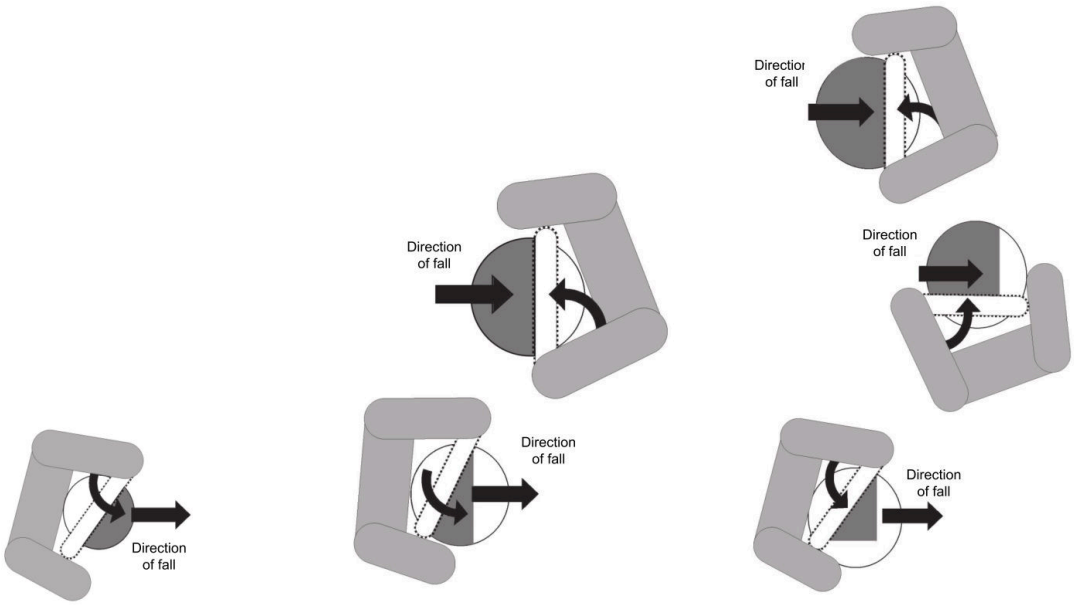


Fig. 7.4 Sailient points of a Waratah felling head [20]



(a) Single cut (b) Double cut (c) Triple cut

Fig. 7.5 Three types of mechanical felling methods [114]

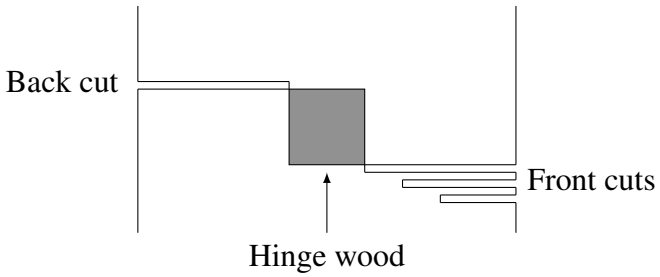


Fig. 7.6 Generation of a Humboldt face with the chainsaw mechanism

7.2.3 Proposed methodology for robotic felling

The mechanized cutting styles provide no or very little control of the tree fall. The tree-felling machine (Fig. 3.4) is not able to create enough force to hold a tree and guide its direction of fall, therefore existing mechanized styles are not suitable. Hence the motor-manual cutting styles will be adapted as this provides passive control. The felling will be executed with two cuts. The first, a front-cut, will be cut to a depth of $\frac{1}{3}$ of the tree diameter. If delayed face closure is required, multiple front-cuts can be executed to create a bigger kerf face [20]. For example a Humboldt face could be imitated as in Fig. 7.6. The second cut, the back-cut, will be executed at a height of $\frac{1}{10}$ of the tree diameter or 12.5 mm above the front-cut, whichever is greater. The back-cut depth will be such that a hinge wood of $\frac{1}{10}$ of the tree diameter, but at least 12.5 mm, remains. It is intended that this procedure will give control over the direction and prevent kick-back.

7.3 Design of novel chainsaw mechanism

In order to extract the maximum volume of wood, the chainsaw is mounted below the gripper as shown in Fig. 7.7. The mechanism pivots with the gripper so that it can be positioned. A cut is generated by moving the chainsaw carriage into the tree using a ball-screw system. Ball-screws were selected as they are widely used for positioning tasks to reach high precision [115]. Detected data input to the felling controller includes bar position, force on bar and tree diameter. The user supplies the cut type (back-cut or front-cut) to the controller.

7.3.1 Mechanical design

The chainsaw module can be divided in two sub-assemblies: The ball-screw drive (Figs. 7.8a and 7.8b) and the chainsaw carriage (Fig. 7.8c). The ball-screw drive consists of two parallel ball-screws, driven by a brushed DC motor over a synchronous belt drive. Rotation of

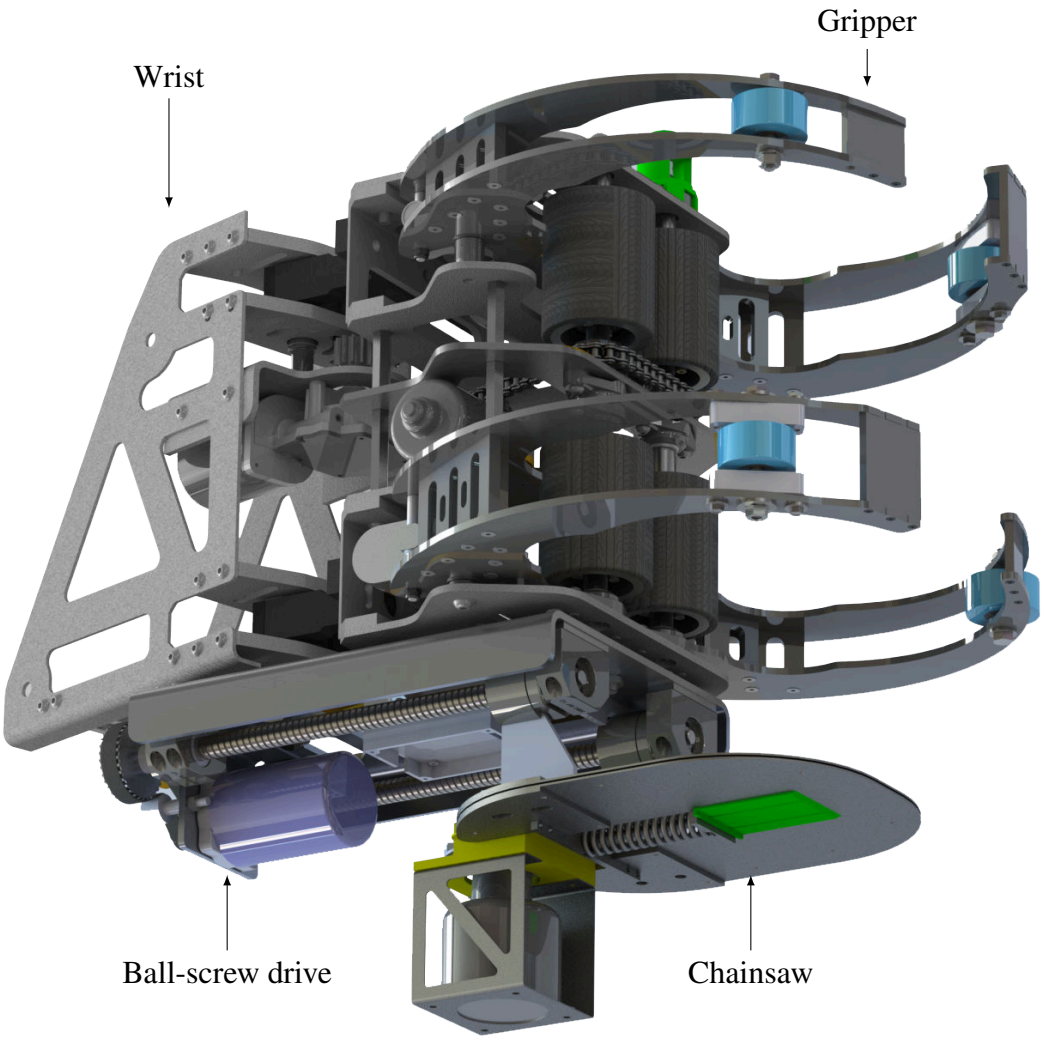


Fig. 7.7 Chainsaw mechanism mounted to the gripper

the ball-screws causes the carriage, connected by two ball nuts, to move forth and back transversally.

The carriage consists of the chainsaw blade designed for this application and a brushless DC motor used to drive the chain. In the middle of the blade a wedge system is placed, which shall prevent the tree clamping the blade or chain.

7.3.2 Electrical design

The system architecture is developed to integrate with Robot Operating System (ROS), which controls the rest of the robotic platform. Position feedback of the ball-screw drive is gained by measuring the ball nut position with a linear resistive stripe. Chain speed is controlled by the Electronic Speed Controller.

A force sensitive resistor (FSR) is integrated in the nose of the blade. This FSR shall detect the vertical force from the trunk pinching the blade. The FSR will be used to gauge the lean of the tree. The moment of release for the gripper will be calculated by tracking the force and position of the blade. The force will also be used as feedback to adjust the procedure whilst cutting, so as to fell the tree in the desired direction.

7.4 Control of the carriage system

In this section the position control of the chainsaw blade is presented. A block diagram of the actuation system is shown in Fig. 7.9. First a model for the ball-screw module will be derived. Based upon the model, a state-space controller will be designed.

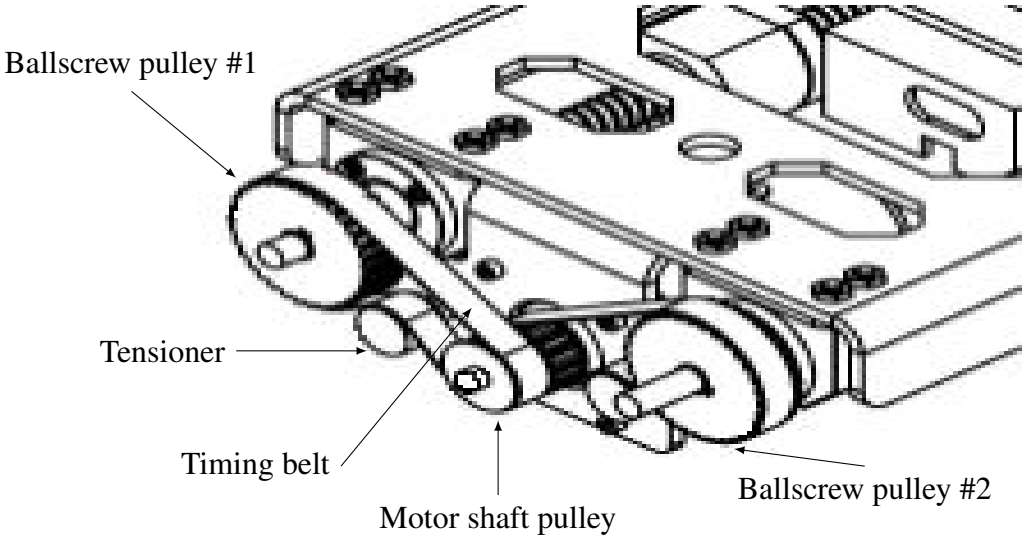
7.4.1 Model derivation

In the following, each block will be described mathematically and the model derived. A rigid model of the ball-screw system is used as the required accuracy is within 0.01 m. All parameters for the system model are summarised in Table 7.1.

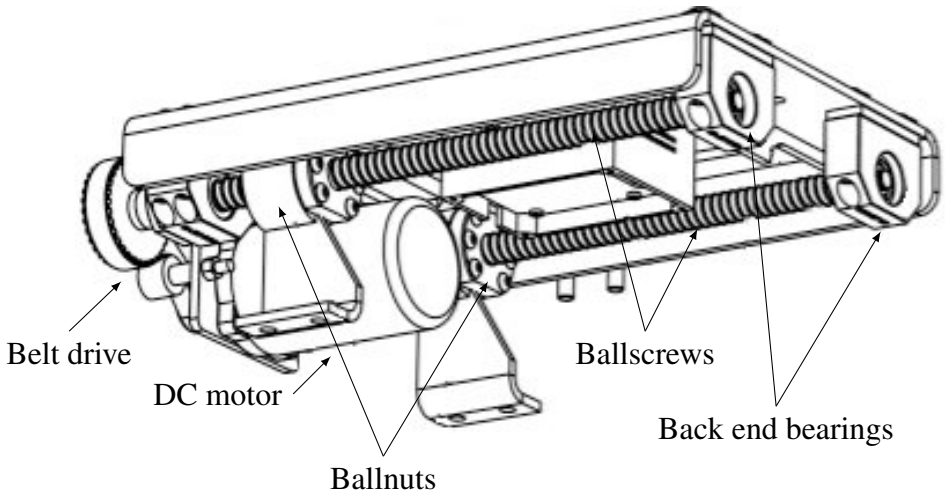
The motor driver sets the supply voltage v in accordance with the control variable, u. The motor driver is described as

$$v(t) = \frac{12}{128} \cdot u(t) \tag{7.1}$$

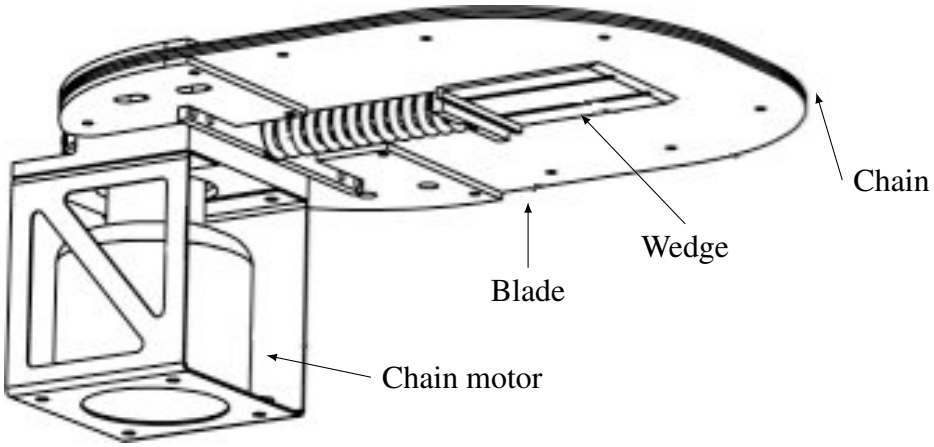
Input to the motor is the voltage over the armature v, output is torque τ and angular velocity ω of the shaft. The armature resistance and inductance are represented by R and L respectively. The armature will develop a current i and have a motor voltage constant k_e. It



(a) Belt drive connection betweem the motor and ball-screws



(b) Underside view of carriage drive



(c) Underside view of chainsaw

Fig. 7.8 Design of chainsaw mechanism

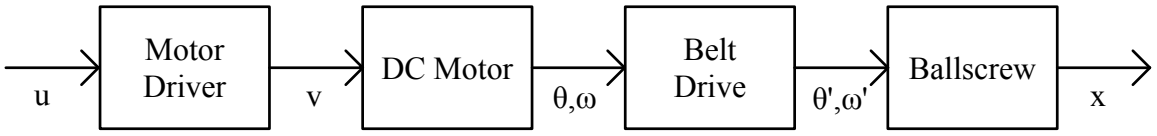


Fig. 7.9 Block diagram of the plant. The variables symbolise in- and outputs, their meaning is explained in the text.

can be shown that

$$L \cdot \frac{d}{dt} i(t) = \frac{12}{128} \cdot u(t) - R \cdot i(t) - k_e \cdot \omega(t)$$

Dynamics of the electric system are assumed to be much faster than the mechanical system dynamics. The electrical control system operates at 100 Hz, faster than relevant mechanical dynamics. It then holds for the armature current that

$$i(t) = \frac{12}{128 \cdot R} \cdot u(t) - \frac{k_e}{R} \cdot \omega(t) \tag{7.2}$$

Torque of the permanent magnet DC motor, with torque constant k_t, can be found as

$$\tau(t) = \frac{12 \cdot k_t}{128 \cdot R} \cdot u(t) - \frac{k_t k_e}{R} \cdot \omega(t) \tag{7.3}$$

The belt drive is depicted in Fig. 7.8a. Two timing belts are used to synchronously drive the ball-screws. For the angular velocity and torque it holds with Eqs. (7.2) and (7.3) that

$$\begin{aligned} \omega_c(t) &= \frac{1}{g} \cdot \omega(t), \\ \tau_m(t) &= \frac{12 \cdot g k_t}{128 \cdot R} \cdot u(t) - \frac{g^2 k_t k_e}{R} \cdot \omega_c(t) \end{aligned} \tag{7.4}$$

The ball-screw drive converts the motor rotation into translational motion for the carriage. With θ_c being the angular position and x the displacement of the carriage it holds that

$$\begin{aligned} x &= \frac{l}{2\pi} \cdot \theta_c, \\ \Rightarrow \dot{x} &= \frac{l}{2\pi} \cdot \omega_c, \\ \Rightarrow \ddot{x} &= \frac{l}{2\pi} \cdot \dot{\omega}_c \end{aligned} \tag{7.5}$$

with $\dot{\theta}_c = \omega_c$. The torque is converted into a force F on the carriage according to

$$\tau_l(t) = \frac{l}{2\pi} \cdot F(t) \tag{7.6}$$

The moment of inertia on the side of the belt drive is determined by the inertia of the electric motor J_M and belt drive J_{BD} as well as the angular acceleration $\dot{\omega}$. This must equate to the sum of all torques. Torques are those of the: motor τ_m, ball-screw/carriage load τ_l and friction τ_v in the motor and belt drive.

$$(J_M + J_{BD}) \cdot \dot{\omega}_c(t) = \tau_m(t) - \tau_l(t) - \tau_v(t) \tag{7.7}$$

For analysis Coulomb, viscous and rolling friction are lumped together under viscous friction and assumed linear. μ_1 is the combined viscous friction coefficient of the motor and belt drive.

$$\tau_v(t) = \mu_1 \cdot \omega_c(t) \tag{7.8}$$

Inserting Eqs. (7.6), (7.8) and (7.4) into Eq. (7.7) and changing coordinates from rotational to translational by Eq. (7.5)

$$\begin{aligned}(J_M + J_{BD}) \cdot \frac{2\pi}{l} \cdot \ddot{x}(t) = &\frac{12 \cdot g k_t}{128 \cdot R} \cdot u(t) - \frac{l}{2\pi} \cdot F(t) \\ &- \left(\frac{g^2 k_t k_e}{R} - \mu_1 \right) \cdot \frac{2\pi}{l} \cdot \dot{x}(t)\end{aligned} \tag{7.9}$$

The carriage subassembly consists of the blade, chain, motor and motor driver with a total mass m. The carriage position, x, is to be controlled. F_f is the viscous friction force of the ball-screw drive. As with the ball-screw system only linear viscous friction is considered (μ_2). It can be shown that

$$m \cdot \ddot{x}(t) + \mu_2 \cdot \dot{x}(t) = F(t) \tag{7.10}$$

The complete equation of motion of the ball-screw drive is derived as

$$\alpha_1 \cdot \ddot{x}(t) + \alpha_2 \cdot \dot{x}(t) = \beta \cdot u(t) \tag{7.11}$$

where,

$$\begin{aligned}\alpha_1 &= (J_M + J_{BD}) \cdot \frac{2\pi}{l} + \frac{lm}{2\pi}, \\ \alpha_2 &= \frac{2\pi g^2 k_t k_e}{Rl} + \frac{l\mu_2}{2\pi} + \frac{2\pi\mu_1}{l}, \\ \beta &= \frac{12 \cdot g k_t}{128 \cdot R}\end{aligned}$$

As the velocity is not measured, a full-state feedback controller is selected and combined with an observer to estimate the velocity of the ball-screw drive. The structure of this controller is shown in Fig. 7.10.

7.5 Implementation for proposed felling policy

Closed-loop control to position the chainsaw blade is established previously. Control is now designed to coordinate the actuators in order to perform the felling process as defined in Section 7.2.3.

7.5.1 The Algorithm and Implementation

The felling procedure from Section 7.2.3 can be divided in two steps, the front and the back cut. The algorithm executes the front or back cut by going through the phases in Algorithm 3.

Algorithm 3 Procedure of the felling control

1. Move saw to 0.02 m before trunk.
2. Accelerate chain to cutting speed.
3. Advance in steps, to allow the chain to return to cutting speed between cuts.
4. Withdraws from the cut when cutting depth achieved.
5. Stop chain.
6. Move back to retracted position.

The motor driver uses an internal feedback loop to maintain chain velocity. As the chainsaw progresses through the tree, the chain velocity decreases due to increasing friction. However, if the chainsaw moves too fast into the tree the limitation of this control is reached, the chain slows down and can result in a stall. To prevent the chain from stopping, the blade does not move continuously into the tree. By stopping at a fixed interval, the chain returns to full cutting speed whilst also clearing swarf from the cut.

As the gripper is above the chainsaw, the time of release during cutting must be determined. Thresholding the force falling edge is used to trigger the gripper release.

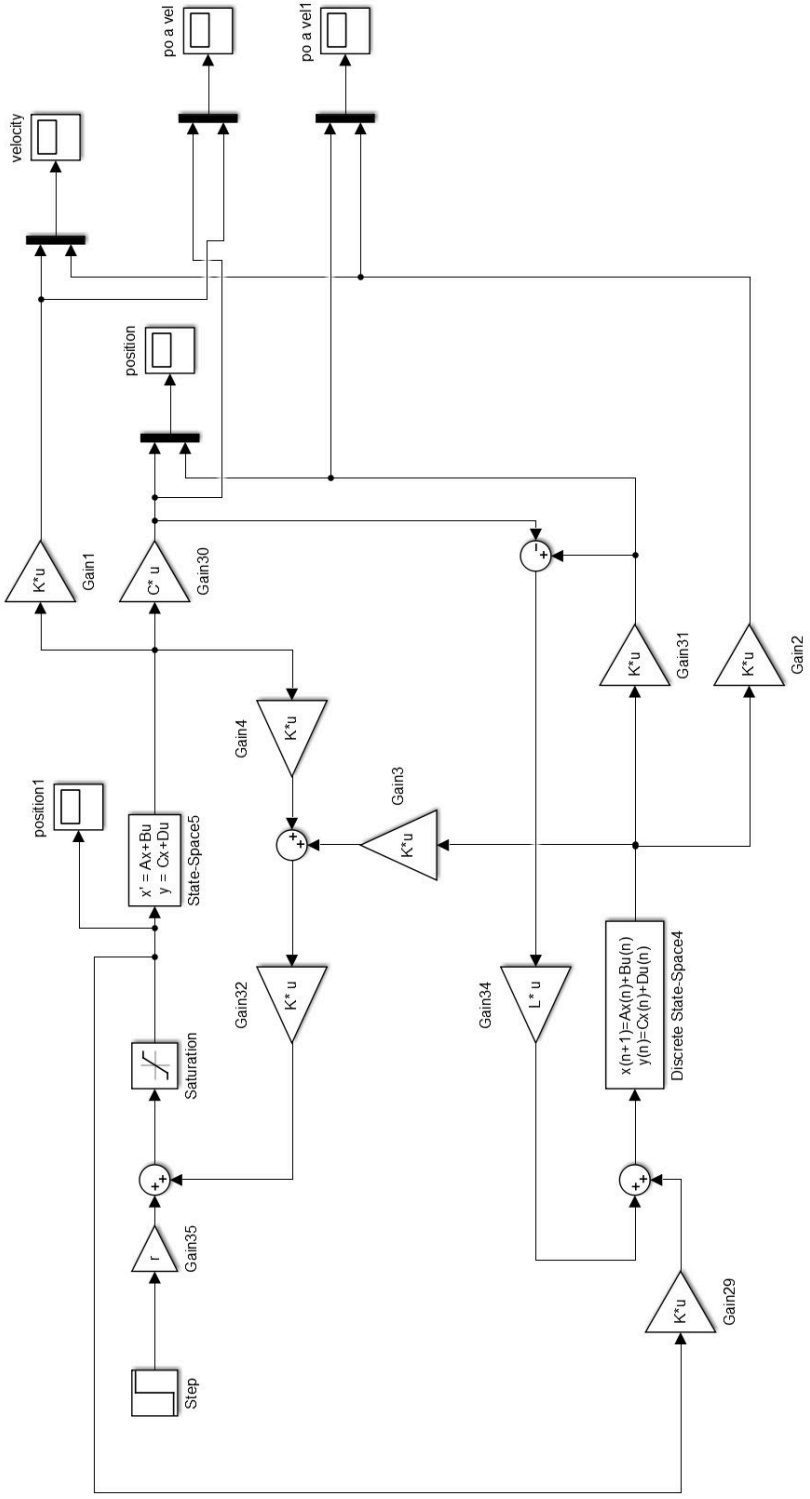


Fig. 7.10 Structure of felling controller developed in Simulink

Table 7.1 List of model parameters

Parameter	Symbol	Unit	Value
Inertia DC motor	J_M	$[\mathrm{kg\,m^2}]$	$9.44 \cdot 10^{-5}$
Inertia belt drive	J_{BD}	$[\mathrm{kg\,m^2}]$	$2.55 \cdot 10^{-4}$
Ball-screw lead	l	$[\mathrm{m}]$	0.005
Weight carriage	m	$[\mathrm{kg}]$	4.480
Gear ratio	g	$[]$	$\frac{32}{14}$
Voltage constant	k_e	$[\mathrm{V\,s\,rad^{-1}}]$	0.02385
Torque constant	k_t	$[\mathrm{N\,m\,A^{-1}}]$	0.02106
Friction coefficient	μ_1	$[\mathrm{kg\,m^2\,s^{-1}}]$	$2.7 \cdot 10^{-3}$
Friction coefficient	μ_2	$[\mathrm{kg\,s^{-1}}]$	6161
Armature resistance	R	$[\Omega]$	0.7

7.6 Results and analysis

The performance of the carriage position controller was gathered during tests into 'open-air', not felling. The felling algorithm was test on trees in the laboratory. Trees were of dry wood and a diameter between 0.15 m and 0.22 m.

7.6.1 Position controller

The loading parameter is found experimentally. The step response was recorded three times for each $u \in [20, 80]$. The error between recorded data and modeled step response was then minimized by unconstrained nonlinear optimization of μ in the model from Eq. (7.11). The quartic polynomial in Eq. (7.12) with parameters in Table 7.2 is used to model μ with coefficient of determination $R^2 = 0.997$

$$\mu_1 = \alpha_4 u^4(t) + \alpha_3 u^3(t) + \alpha_2 u^2(t) + \alpha_1 u(t) + \alpha_0 \tag{7.12}$$

The step response of the closed-loop ball-screw drive is shown in Fig. 7.12. This step was performed with a command variable of $u = 40$, for a step size of 0.1 m. The set-point is achieved with an offset of approximately 0.2 mm, equivalent to the position sensor accuracy.

Table 7.2 List of coefficients for model in Eq. (7.12)

Coefficient:	α_4	α_3	α_2	α_1	α_0
Value:	6.79×10^{-3}	-1.56	133	-4940	7.31×10^4

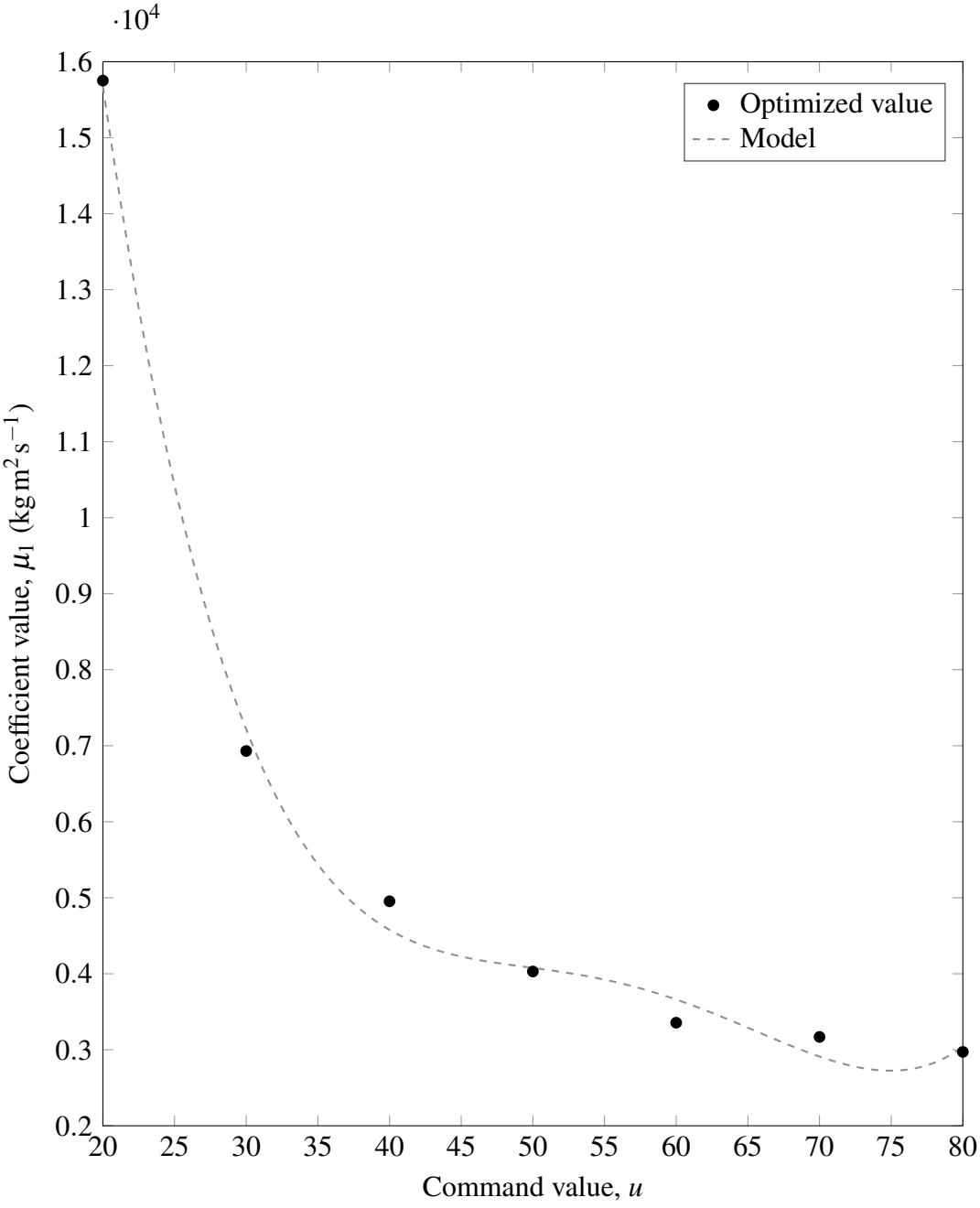


Fig. 7.11 Behaviour and model of reactive force coefficient μ_1

No overshooting occurs within the resolution of the position sensor as desired by the control strategy. This prevents the blade overshooting into the tree during felling.

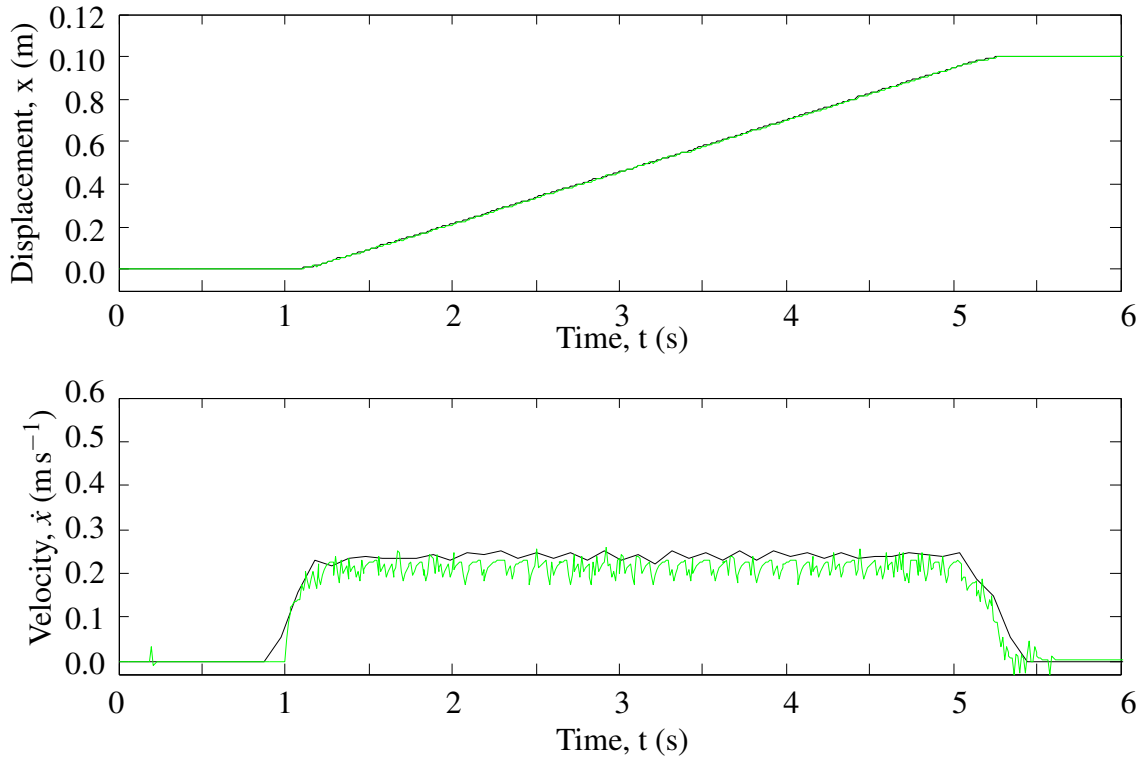


Fig. 7.12 Step response of the plant showing measured position and velocity (black) and estimated states by the observer (green)

7.6.2 Felling control

Felling in the natural direction of fall was successfully executed, as Fig. 7.13 shows. However with the original width of the hinge wood of $1/10^{th}$ of the tree diameter the tree did not fall in all cases. The hinge wood width had to be reduced until front and back cut were overlapping and the hinge wood was reduced further. Results are improved by continuing the back cut until the FSR recognizes that the tree is starting to fall. By using the FSR in this manner, diversity in tree stiffness/density is not a limiting factor in hinge functionality.

During the test series kick-back did not occur due to the height difference between front and back cut. Therefore the proposed felling methodology was found to be suitable when combined with the FSR.

(a) Bottom piece (“stump”)

(b) Top piece (“log”)

Fig. 7.13 Results of a cut performed by the chainsaw module

7.6.3 Force sensing

In Figs. 7.14 and 7.15 the data measured during the felling process is shown, with key elements described in Table 7.3. The force on the chainsaw bar increases until the tree starts to lean over into the front-cut (region b). After reaching a maximum force on the bar, the force deceases as the tree begins to lean over in the front-cut (region c). This allows the fall of the tree to be monitored and the required depth of front cut to be updated on-line as required. Once the critical amount of wood has been cut, the tree begins to fall uncontrolled resulting in the rapid decrease in force from $t = 48\,\mathrm{s}$ onward (region d).

The blade consumed 29 s to fell the tree. Based upon the duration spent in region c, the gripper had 10 s in which to release the tree is the decision is based upon the FSR alone. The fall of the tree can also be estimated on position, from knowledge of the amount of wood left in the back-cut. By combining the knowledge of both the FSR and hence tree state, as well as current position of the back-cut, the moment of release can be calculated. The latter approach allows more slack-time in the system, in this case as much as 23 s. As the chainsaw can continue cutting without the need for the gripper to be locked to the tree, an early departure provides a safer option. Leaving early does however increase the probability of failures during cutting, these are safer than the robot releasing too late and therefore acceptable.

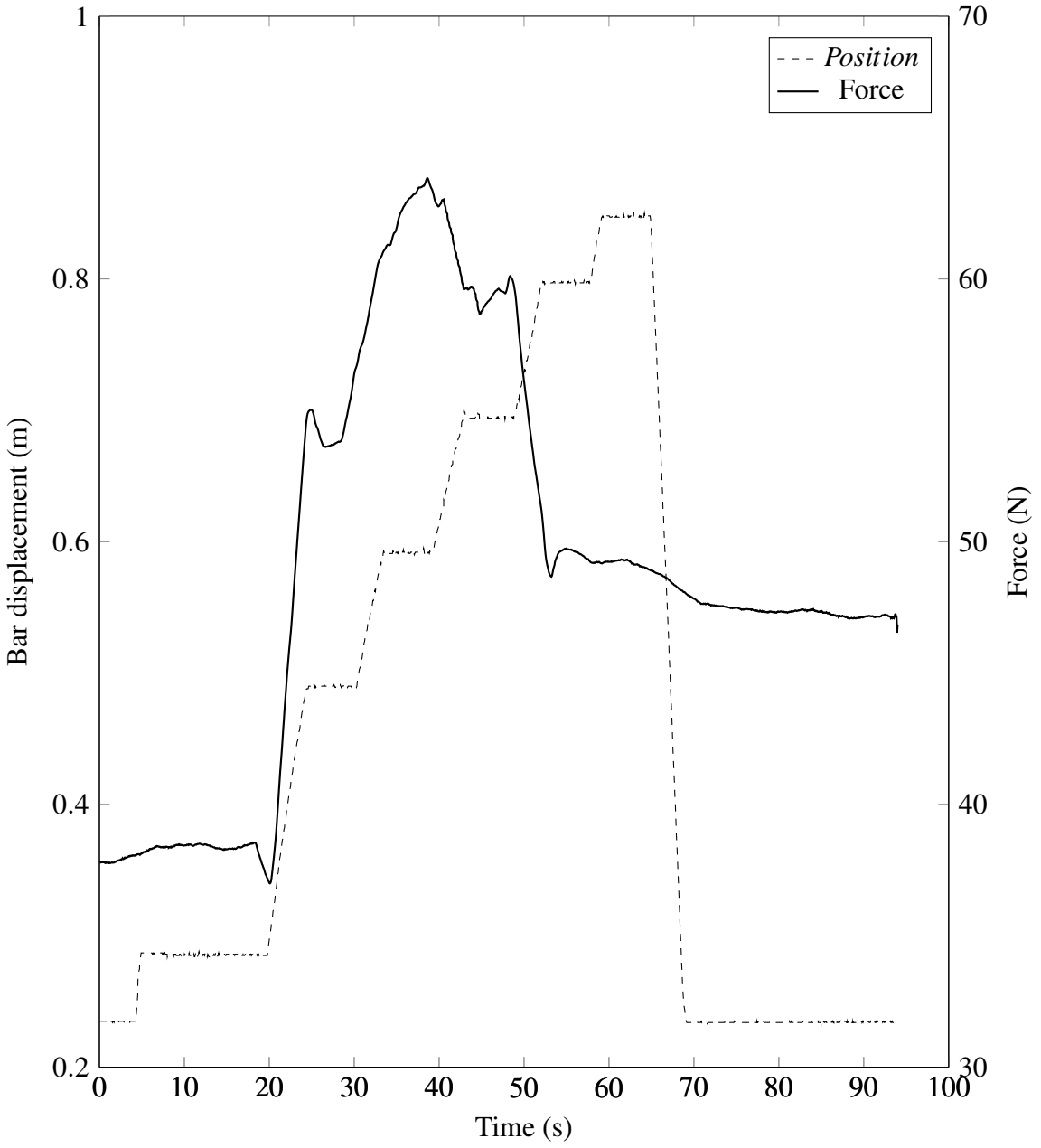


Fig. 7.14 Chainsaw position and force during felling

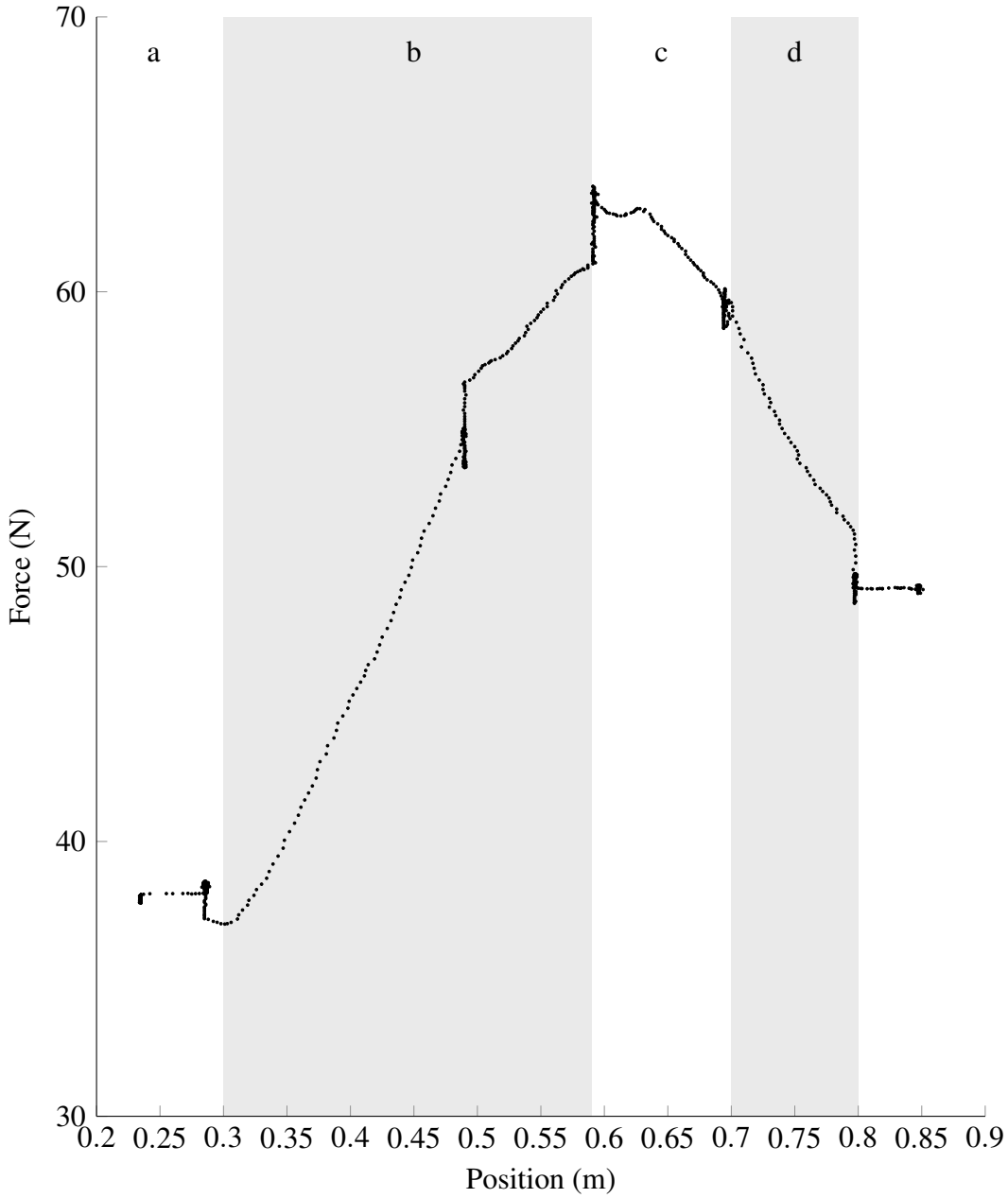


Fig. 7.15 Scatter of forces on chainsaw bar as a function of position

Table 7.3 Description of events in FSR data during felling (Fig. 7.14)

Time domain (s)	Position (m)	Event description
$[\mathbf{0}, \mathbf{19})$	$[\mathbf{0.2}, \mathbf{0.3})$	The blade moves toward the tree.
19	**0.3**	Chainsaw makes first contact with the tree.
$(\mathbf{19}, \mathbf{38})$	$[\mathbf{0.3}, \mathbf{0.6})$	Chainsaw continues through the tree. Force increases as tree leans onto bar.
38	**0.6**	Force peaks. Cut is now sufficiently deep to cause the tree to start moving into front cut.
$(\mathbf{38}, \mathbf{45})$	$[\mathbf{0.6}, \mathbf{0.7})$	Tree slowly leans over into the front cut, releasing the chainsaw bar.
45	**0.7**	The latest time at which the robot can release the tree.
$[\mathbf{48}, \mathbf{53}]$	$[\mathbf{0.7}, \mathbf{0.8})$	Force rapidly decreases as tree falls overs uncontrolled.
$(\mathbf{53}, \mathbf{93}]$	$[\mathbf{0.8}, \mathbf{0.85})$	FSR returns to a no-load position but experiences hysteresis.

7.7 Conclusion

A novel cutting mechanism and suitable cutting technique has been developed. To realize this system the chainsaw module was automated using two controllers. A model for the ball-screw drive was derived to achieve closed-loop position control for the carriage, a positioning accuracy of approximately 0.2 mm was accomplished. A supervisory felling controller was designed and implemented to run the position controller. The novel inclusion of a sensor to measure downward force on the chainsaw allowed detection of the tree state. The chainsaw module with the felling control was successfully tested in a laboratory environment.

Future work would include the optimization of the blade speed control so as to reduce the time needed for felling. By adding speed feedback for the chain, the carriage velocity can be used to maintain the target saw speed. This would remove the present need to pause carriage advancement whilst the chain regains speed. Depending on the tree's mechanical properties, a different cutting depth is necessary to initiate the tree fall, a static depth for the back cut does not always fall tree. Therefore further integration of the force and position data along with external data gathered by the robot's LiDAR system may aid to improve control in felling.

Chapter 8

Conclusions and Future work

8.1 Conclusion

In this work, a teleoperated robot is presented with the capability to fell trees on steep terrain. Using a form of brachiation, the robot can maneuver through the forest without needing to touch the ground. Controlled over Wi-Fi by an operator in a safe area, the machine achieves the goal of increasing personal safety during tree felling.

From the literature (Chapter 2) it was seen that there are many robotics platforms in agriculture. It was also found however that no suitable robotic platform had been developed for purpose of felling of trees, let alone on New Zealand's steep terrain forest. It was also shown that many of the requirements for robotic operation in forest, such as object detection and route planning, were not met by existing literature.

In order to address the needs of industry for such a machine, the developed of a suitable platform was shown in Chapter 3. The resultant machine contains 11 degrees of freedom and weighing 75 kg.

A kinematic and dynamic analysis was performed in Chapter 4. From the kinematics, identification of the workspace of the end effector was possible. Within the workspace, the Yoshikawa manipulability index was used to identify the manipulability at each point and found the optimal grasping point for the end effector. Understanding of the workspace allows for better control of the machine within its boundary conditions. The dynamic model was used to find inertia, torque and acceleration at each joint. It was seen how pose affects inertia and how this can be manipulated to minimise the required effort and strain. Actuator performance was characterized by the joint torque which identified the areas of low capacity

in the workspace.

An algorithm has been developed to generate routes for autonomous harvesting robots. Termed 'k-Means Partitioned Space Path Planning', or KPSPP. The proposed algorithm successfully planned coverage routes through forests containing up to 36 000 trees, demonstrating applicability to commercially sized plantations. Coverage planning of discretized crops in two- or three-dimensions had not yet been available. Trials involving 14 real-world forests were used in benchmarking. In the tests, the KPSPP algorithm out-performed all other algorithms in large forest datasets. Performance of the proposed method at lower set sizes was less competitive however. Despite the decreased performance in sets with less than 1000 trees, was still rated as performing equivalently as other algorithms tested.

In order to recognize Pine trees in plantation forest, a system was devoped consisting of a sensing rig and recognition algorithm. An automatic panning system is developed, rotating a two-dimensional LiDAR unit resulting in a three-dimensional field of view. The system requires only a single scan, taken automatically, from a terrestrial LiDAR unit mounted on-board the robot, in order to classify objects. An algorithm is developed to process the resultant point cloud to identify trees, ground and other objects. Trees are found in a novel way, using the natural tendency of trees to grow parallel with the gravity vector allowing a search for faces that have horizontal normal vectors. Compared to other results in the literature the algorithm is able to identify trees in diverse, cluttered environments using less scans than others. The detection system also achieved RMS errors that improved the state of the art by an order of magnitude.

A novel cutting mechanism and suitable cutting technique has been developed. Firstly an electric chainsaw was designed and manufactured. This chainsaw was mounted on a carriage and ball-screws were used to position it throughout the cutting process. A model for the ball-screw drive was derived to achieve closed-loop position control for the carriage. A supervisory felling controller was designed and implemented to run the position controller. This lead to the automation of the chainsaw to perform felling according to a felling procedure. The procedure was developed from existing mechanical methods and modified for the robot system in Chapter 3. The novel inclusion of a sensor to measure downward force on the chainsaw allowed detection of the tree state. By measuring the tree state, timing for the felling procedure was derived. The chainsaw module with the felling control was successfully tested in a laboratory environment using real tree trunks.

8.2 Future work

The control strategy can be improved such that it is aware of the machine properties identified in Chapter 4. Particularly grasping trunks in the correct orientation to facilitate felling directly. Poses leading to optimum actuator performance or grasp can be used to optimize long term performance of the machine for energy efficiency or speed.

The performance of KPSPP in sets of less than 1000 trees is to be improved. This can be achieved by reformulation of the partitioning strategy to more suitably divide the planning space. Route planning of the machine though forest can also benefit from the detailed model of workspace manipulability, gained in Section 4.2. Integration of this information would lead to optimal grasping occurring more frequently whilst also maximizing advantages in kinematic redundancy.

Fusion of optical camera data with LiDAR point cloud data can be used to increase quality in point characterization. If points can be more accurately identified as being trunk, ground, forest clutter or other then the rate of false positives and negatives can be improved. Piecewise cylinder models along the trunk should also be employed so as to estimate it's surface more accurately. To aid direction felling, an understanding of the tree's center of mass should be achieved. This can be done using LiDAR data to estimate the canopy distribution and lean of tree.

Future work on the felling capability would include the optimization of the blade speed control so as to reduce the time needed for felling. By adding speed feedback for the chain, the carriage velocity can be used to maintain the target saw speed. This would remove the present need to pause carriage advancement whilst the chain regains speed. Depending on the tree's mechanical properties, a different cutting depth is necessary to initiate the tree fall, a static depth for the back cut does not always fall tree. Therefore further integration of the force and position data along with external data gathered by the robot's LiDAR system may aid to improve control in felling.

Bibliography

[1] Department of Labour. *A Guide to Safety in Tree Felling and Cross Cutting*. Occupational Safety and Health Service, Department of Labour, New Zealand, 1980.

[2] Statistics New Zealand. *Exports - Top 20 commodities*. Accessed: 27th Feb 2015. 2013. URL: https://www.nzte.govt.nz/en/invest/statistics/.

[3] New Zealand Forest Owners Association. *New Zealand plantation forest industry: Facts & Figures*. 2011.

[4] *Situation and Outlook for New Zealand Agriculture and Forestry*. Report. Ministry of Agriculture and Forestry, 2011.

[5] *Provisional estimates of tree stock sales and forest planting in 2013*. Report. Ministry for Primary Industries, 2014.

[6] Keith Raymond. "Innovation to increase profitability of steep terrain harvesting in New Zealand". In: *New Zealand Journal of Forestry* 57.2 (2012), pp. 19–23.

[7] Occupational Safety and Health Service. *Approved Code of Practice for Safety and Health in Forest Operations*. 1999.

[8] D. Amishev, T. Evanson, and K. Raymond. "Felling and Bunching on Steep Terrain - A Review of the Literature". In: *FFR Technical Note* 1.7 (2009), pp. 1–10.

[9] K. Raymond. "Innovative Harvesting Solutions: A Step Change". In: *New Zealand Journal Of Forestry* 55 (2010), pp. 4–9.

[10] T. Evanson and D. Amishev. "A New Method for Bunching Trees on Steep Terrain". In: *FFR Technical Note* 2.5 (2009), pp. 1–9.

[11] T. Evanson and D. Amishev. "Productivity Impacts of Bunching for Hauler Extraction". In: *FFR Technical Note* (2009), pp. 1–9.

[12] K. M. Bayne and R. J. Parker. "The introduction of robotics for New Zealand forestry operations: Forest sector employee perceptions and implications". In: *Technology in Society* 34 (2012), pp. 138–148.

[13] James Manyika et al. *Disruptive technologies: Advances that will transform life, business, and the global economy*. Report. McKinsey Global Institute, 2013.

[14] Department of Labour. *Forestry Sector Action Plan 2010-2013*. Department of Labour, 2011.

[15] Nancy Swarbrick. *'Waikato places - Tokoroa', Te Ara - the Encyclopedia of New Zealand*. 6th Jan 2015. 2013.

[16] Ministry of Forestry and New Zealand Logging Industry Research Organisation. *Harvesting a small forest*. Vol. 7. Small Forest Management. Ministry of Forestry, 1996.

[17] T. Nisbet, J. Dutch, and A. Moffat. *Whole-Tree Harvesting: A Guide to Good Practice*. Forestry Authority, 1997.

[18] Deere & Company. *909MH Tracked Harvester*. Accessed: 27th Feb 2015. 2015. URL: https://www.deere.com/en_US/products/equipment/harvesters/tracked_harvesters/909mh/909mh.page?.

[19] Competenz. *Best practice guidelines for Cable Logging*. 2000.

[20] Competenz. *Best practice guidelines for Mechanised Harvesting and Processing*. 2002.

[21] Tim A. Bentley, Richard J. Parker, and Liz Ashby. "Understanding felling safety in the New Zealand forest industry". In: *Applied ergonomics* 36.2 (2005), pp. 165–175.

[22] D. F. Scott. "A study of logger fatalities from 1992-2000". In: *Injury Prevention* 10.4 (2004), pp. 239–243. DOI: 10.1136/ip.2003.004663.

[23] E. Roman-Jordan P. M. Albizu-Urionabarrenetxea E. Tolosana-Esteban. "Safety and health in forest harvesting operations. Diagnosis and preventive actions. A review". In: *Forest Systems* 22.3 (2013). DOI: 10.5424/fs/2013223-02714.

[24] ClimbMax Equipment Ltd. *ClimbMax Steep Slope Harvester*. Accessed: 25th Feb 2014. 2014. URL: http://www.climbmax.co.nz/\#!/portfolio-view/climbmax-steep-slope-harvester.

[25] Ministry for Primary Industries. *Steepland Harvesting*. Accessed: 17th Jan 2014. 2014. URL: http://www.mpi.govt.nz/agriculture/funding-programmes/primary-growth-partnership/steepland-harvesting.

[26] Mario M. Foglia and Giulio Reina. "Agricultural robot for radicchio harvesting". In: *Journal of Field Robotics* 23.6-7 (2006), pp. 363–377. DOI: 10.1002/rob.20131.

[27] Kanae Tanigaki et al. "Cherry-harvesting robot". In: *Computers and Electronics in Agriculture* 63.1 (2008), pp. 65–72. DOI: 10.1016/j.compag.2008.01.018.

[28] E. J. Van Henten et al. "Optimal manipulator design for a cucumber harvesting robot". In: *Computers and Electronics in Agriculture* 65.2 (2009), pp. 247–257. DOI: 10.1016/j.compag.2008.11.004.

[29] Zhao De-An et al. "Design and control of an apple harvesting robot". In: *Biosystems Engineering* 110.2 (2011), pp. 112–122. DOI: 10.1016/j.biosystemseng.2011.07.005.

[30] Shigehiko Hayashi et al. "Evaluation of a strawberry-harvesting robot in a field test". In: *Biosystems Engineering* 105.2 (2010), pp. 160–171. DOI: 10.1016/j.biosystemseng.2009.09.011.

[31] Tin Lun Lam and Yangsheng Xu. *Tree Climbing Robot: Design, Kinematics and Motion Planning*. Springer Berlin Heidelberg, 2012. DOI: 10.1007/978-3-642-28311-6.

[32] D. P. Soni et al. "Autonomous arecanut tree climbing and pruning robot". In: *Emerging Trends in Robotics and Communication Technologies (INTERACT), 2010 International Conference on*, pp. 278–282. DOI: 10.1109/INTERACT.2010.5706160.

[33] W. Chonnaparamutt et al. “Development of a timberjack-like pruning robot: Climbing experiment and fuzzy velocity control”. In: *ICCAS-SICE, 2009*, pp. 1195–1199.

[34] S. Ueki et al. “Development and experimental study of a novel pruning robot”. In: *Artificial Life and Robotics* 16.1 (2011), pp. 86–89. DOI: 10.1007/s10015-011-0892-1.

[35] Lam Tin Lun and Xu Yangsheng. “Climbing Strategy for a Flexible Tree Climbing Robot”. In: *Robotics, IEEE Transactions on* 27.6 (2011), pp. 1107–1117. DOI: 10.1109/TRO.2011.2162273.

[36] Y. Ishigure, K. Hirai, and H. Kawasaki. “A pruning robot with a power-saving chainsaw drive”. In: *Mechatronics and Automation (ICMA), IEEE International Conference on*. 2013, pp. 1223–1228. DOI: 10.1109/ICMA.2013.6618088.

[37] R. S. Dahiya et al. “Tactile Sensing - From Humans to Humanoids”. In: *Robotics, IEEE Transactions on* 26.1 (2010), pp. 1–20. DOI: 10.1109/TRO.2009.2033627.

[38] S. Chitta et al. “Tactile Sensing for Mobile Manipulation”. In: *Robotics, IEEE Transactions on* 27.3 (2011), pp. 558–568. DOI: 10.1109/TRO.2011.2134130.

[39] T. Maeno, T. Kawamura, and Cheng Sen-Chieh. “Friction estimation by pressing an elastic finger-shaped sensor against a surface”. In: *Robotics and Automation, IEEE Transactions on* 20.2 (2004), pp. 222–228. DOI: 10.1109/TRA.2003.820850.

[40] Chuang Cheng-Hsin et al. “Flexible tactile sensor for the grasping control of robot fingers”. In: *Advanced Robotics and Intelligent Systems (ARIS), 2013 International Conference on*, pp. 141–146. DOI: 10.1109/ARIS.2013.6573549.

[41] Zheng Yu. “An Efficient Algorithm for a Grasp Quality Measure”. In: *Robotics, IEEE Transactions on* 29.2 (2013), pp. 579–585. DOI: 10.1109/TRO.2012.2222274.

[42] A. Arms et al. “Arboreal locomotion in small New-World monkeys”. In: *Zeitschrift fur Morphologie und Anthropologie* 83.2/3 (2002), pp. 243–263. DOI: 10.2307/25757608.

[43] James R. Usherwood and John E. A. Bertram. “Understanding brachiation: insight from a collisional perspective”. In: *Journal of Experimental Biology* 206.10 (2003). 10.1242/jeb.00306, pp. 1631–1642.

[44] James R. Usherwood, Susan G. Larson, and John E. A. Bertram. “Mechanisms of force and power production in unsteady ricochetal brachiation”. In: *American Journal of Physical Anthropology* 120.4 (2003), pp. 364–372. DOI: 10.1002/ajpa.10133.

[45] J. E. A. Bertram et al. “A point-mass model of Gibbon Locomotion”. In: *The Journal of Experimental Biology* 202 (1999), pp. 2609–2617.

[46] Young-Hui Chang, John E. A. Bertram, and David V. Lee. “External forces and torques generated by the brachiating white-handed gibbon (Hylobates lar)”. In: *American Journal of Physical Anthropology* 113.2 (2000), pp. 201–216. DOI: 10.1002/1096-8644(200010)113:2<201::AID-AJPA5>3.0.CO;2-S.

[47] Toshio Fukuda and Fuminori Saito. “Motion control of a brachiation robot”. In: *Robotics and Autonomous Systems* 18.1 (1996), pp. 83–93. DOI: 10.1016/0921-8890(96)00002-4.

[48] Ali Meghdari et al. “Minimum control effort trajectory planning and tracking of the CEDRA brachiation robot”. In: *Robotica* 31.07 (2013), pp. 1119–1129. DOI: 10.1017/S0263574713000362.

[49] J. Nakanishi, T. Fukuda, and D. E. Koditschek. “A brachiating robot controller”. In: *Robotics and Automation, IEEE Transactions on* 16.2 (2000), pp. 109–123. DOI: 10.1109/70.843166.

[50] H. Nishimura and K. Funaki. “Motion control of three-link brachiation robot by using final-state control with error learning”. In: *Mechatronics, IEEE/ASME Transactions on* 3.2 (1998), pp. 120–128. DOI: 10.1109/3516.686680.

[51] Toshio Fukuda et al. *Multi-Locomotion Robotic Systems: New Concepts of Bio-inspired Robotics*. Springer Tracts in Advanced Robotics. Springer Berlin Heidelberg, 2012. DOI: 10.1007/978-3-642-30135-3.

[52] Jian Jin and Lie Tang. “Coverage path planning on three-dimensional terrain for arable farming”. In: *Journal of Field Robotics* 28.3 (2011), pp. 424–440. DOI: 10.1002/rob.20388.

[53] I. A. Hameed. “Intelligent Coverage Path Planning for Agricultural Robots and Autonomous Machines on Three-Dimensional Terrain”. In: *Journal of Intelligent & Robotic Systems* 74.3-4 (2014), pp. 965–983. DOI: 10.1007/s10846-013-9834-6.

[54] Ibrahim A. Hameed, Dionysis Bochtis, and Claus A. Sorensen. “An Optimized Field Coverage Planning Approach for Navigation of Agricultural Robots in Fields Involving Obstacle Areas”. In: *Int J Adv Robot Syst* 10.231 (2013). DOI: 10.5772/56248.

[55] D. Bochtis et al. “Route planning for orchard operations”. In: *Computers and Electronics in Agriculture* 113.0 (2015), pp. 51–60. DOI: 10.1016/j.compag.2014.12.024.

[56] D. D. Bochtis and C. G. Sorensen. “The vehicle routing problem in field logistics part I”. In: *Biosystems Engineering* 104.4 (2009), pp. 447–457. DOI: 10.1016/j.biosystemseng.2009.09.003.

[57] D. Bochtis et al. “Design of a Wildlife Avoidance Planning System for Autonomous Harvesting Operations”. In: *Int J Adv Robot Syst* 11.6 (2014). DOI: 10.5772/57442.

[58] Bo-Yeong Kang et al. “ROBIL: Robot Path Planning Based on PBIL Algorithm”. In: *Int J Adv Robot Syst* 11.147 (2014), pp. 1–14. DOI: 10.5772/58872.

[59] Jiatong Bao, Hongru Tang, and Aiguo Song. “Combining Vision Learning and Interaction for Mobile Robot Path Planning”. In: *Int J Adv Robot Syst* 9.102 (2012), pp. 1–8. DOI: 10.5772/50827.

[60] David Gleich. *gaimc: Graph Algorithms In Matlab Code*. Computer Program. 2009.

[61] Derek G. Corneil and Richard M. Krueger. “A Unified View of Graph Searching”. In: *SIAM Journal on Discrete Mathematics* 22.4 (2008), pp. 1259–1276. DOI: 10.1137/050623498.

[62] Joseph Kirk. *Traveling Salesman Problem - Nearest Neighbor*. Computer Program. 2008.

[63] ‘Fatime’. *Matlab, More efficient code to find all paths with smaller than specified length between two nodes in sparse graph*. Accessed: 6th Nov 2013. 2013. URL: http://stackoverflow.com/questions/18169680/matlab-more-efficient-code-to-find-all-paths-with-smaller-than-specified-length.

[64] Michael Kohl, Steen Magnussen, and Marco Marchetti. "Remote Sensing". In: *Sampling Methods, Remote Sensing and GIS Multiresource Forest Inventory*. Tropical Forestry. Springer Berlin Heidelberg, 2006. Chap. 4, pp. 197–238. DOI: 10.1007/978-3-540-32572-7_4.

[65] Matti Maltamo, Erik Naesset, and Jari Vauhkonen. *Forestry applications of airborne laser scanning : concepts and case studies*. Dordrecht: Springer, 2014.

[66] Ronald E. McRoberts and Erkki O. Tomppo. "Remote sensing support for national forest inventories". In: *Remote Sensing of Environment* 110.4 (2007), pp. 412–419. DOI: 10.1016/j.rse.2006.09.034.

[67] Kevin Lim et al. "LiDAR remote sensing of forest structure". In: *Progress in physical geography* 27.1 (2003), pp. 88–106. DOI: 10.1191/0309133303pp360ra.

[68] Darius S. Culvenor. "TIDA: an algorithm for the delineation of tree crowns in high spatial resolution remotely sensed imagery". In: *Computers & Geosciences* 28.1 (2002), pp. 33–44. DOI: 10.1016/S0098-3004(00)00110-2.

[69] Zhou Guiyun, Wang Bin, and Zhou Ji. "Automatic Registration of Tree Point Clouds From Terrestrial LiDAR Scanning for Reconstructing the Ground Scene of Vegetated Surfaces". In: *Geoscience and Remote Sensing Letters, IEEE* 11.9 (2014), pp. 1654–1658. DOI: 10.1109/LGRS.2014.2314179.

[70] Yotam Livny et al. "Automatic Reconstruction of Tree Skeletal Structures from Point Clouds". In: *ACM Trans. Graph.* 29.6 (Dec. 2010), 151:1–151:8. DOI: 10.1145/1882261.1866177.

[71] Jean-Francois Cote, Richard A. Fournier, and Richard Egli. "An architectural model of trees to estimate forest structural attributes using terrestrial LiDAR". In: *Environmental Modelling & Software* 26.6 (2011), pp. 761–777. DOI: 10.1016/j.envsoft.2010.12.008.

[72] Hong-wei Liu et al. "The reconstruction of three-dimensional tree models from terrestrial LiDAR". In: *Computer Science and Automation Engineering (CSAE), 2011 IEEE International Conference on*. Vol. 4, pp. 371–374. DOI: 10.1109/CSAE.2011.5952871.

[73] T. Palleja et al. "Sensitivity of tree volume measurement to trajectory errors from a terrestrial LIDAR scanner". In: *Agricultural and Forest Meteorology* 150.11 (2010), pp. 1420–1427. DOI: 10.1016/j.agrformet.2010.07.005.

[74] Matthew W. McDaniel et al. "Terrain classification and identification of tree stems using ground-based LiDAR". In: *Journal of Field Robotics* 29.6 (2012), pp. 891–910. DOI: 10.1002/rob.21422.

[75] R.M. Palenichka and M.B. Zaremba. "Multiscale Isotropic Matched Filtering for Individual Tree Detection in LiDAR Images". In: *Geoscience and Remote Sensing, IEEE Transactions on* 45.12 (2007), pp. 3944–3956. DOI: 10.1109/TGRS.2007.908875.

[76] D. Pont et al. "Calibrated tree counting on remotely sensed images of planted forests". In: *International Journal of Remote Sensing* 36.15 (2015), pp. 3819–3836. DOI: 10.1080/01431161.2015.1054048.

[77] L. Wallace, A. Lucieer, and C.S. Watson. "Evaluating Tree Detection and Segmentation Routines on Very High Resolution UAV LiDAR Data". In: *Geoscience and Remote Sensing, IEEE Transactions on* 52.12 (2014), pp. 7619–7628. DOI: 10.1109/TGRS.2014.2315649.

[78] A.M. Wallace et al. "Design and Evaluation of Multispectral LiDAR for the Recovery of Arboreal Parameters". In: *Geoscience and Remote Sensing, IEEE Transactions on* 52.8 (2014), pp. 4942–4954. DOI: 10.1109/TGRS.2013.2285942.

[79] M. Hussein et al. "Matching of ground-based LiDAR and aerial image data for mobile robot localization in densely forested environments". In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pp. 1432–1437. DOI: 10.1109/IROS.2013.6696537.

[80] W. Ali, F. Georgsson, and T. Hellstrom. "Visual tree detection for autonomous navigation in forest environment". In: *Intelligent Vehicles Symposium, 2008 IEEE*, pp. 560–565. DOI: 10.1109/IVS.2008.4621315.

[81] Cui Jin Qiang et al. "Autonomous navigation of UAV in forest". In: *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*. 2014, pp. 726–733. DOI: 10.1109/ICUAS.2014.6842317.

[82] F. Pinage, J. R. H. Carvalho, and J. Pinheiro de Queiroz Neto. "Natural Landmark Tracking Method to Support UAV Navigation over Rain Forest Areas". In: *Computing System Engineering (SBESC), 2012 Brazilian Symposium on*, pp. 105–110. DOI: 10.1109/SBESC.2012.28.

[83] Rien Visser and Raffaele Spinelli. "Determining the shape of the productivity function for mechanized felling and felling-processing". In: *Journal of Forest Research* 17.5 (2012), pp. 397–402. DOI: 10.1007/s10310-011-0313-2.

[84] Drew Arnold and John Parmigiani. "A method for detecting the occurrence of chainsaw kickback". In: *ASME 2011 International Mechanical Engineering Congress and Exposition*. Vol. 7. 2011, pp. 441–447. DOI: 10.1115/IMECE2011-64459.

[85] I. Osborne, D. Arnold, and G. Hall. *Kickback detection method and apparatus*. Patent. wo2012021752. Feb. 2012.

[86] D. R. Peterson, A. J. Brammer, and M. G. Cherniack. "Exposure monitoring system for day-long vibration and palm force measurements". In: *International Journal of Industrial Ergonomics* 38.9 (2008), pp. 676–686. DOI: 10.1016/j.ergon.2007.10.019.

[87] Mo Li et al. "Biomimeitc Design of a Stubble-Cutting Disc Using Finite Element Analysis". In: *Journal of Bionic Engineering* 10.1 (2013), pp. 118–127. DOI: 10.1016/S1672-6529(13)60206-1.

[88] C.V. Meaclem et al. "Sensor guided biped felling machine for steep terrain harvesting". In: *Automation Science and Engineering (CASE), 2014 IEEE International Conference on*. Aug. 2014, pp. 984–989. DOI: 10.1109/CoASE.2014.6899446.

[89] Richard Parker. *RE: Tree Sizes and Distributions for Peter*. Personal Communication. Mar. 2013.

[90] MathWorks. *Delaunay Triangulation*. Accessed: 4th Nov 2013. 2013. URL: http://www.mathworks.com/help/matlab/math/delaunay-triangulation.html.

[91] George Wareing et al. *Project 21: Biped Felling Machine - Final Year Report*. Report. University of Canterbury, Department of Mechanical Engineering, 2013.

[92] A. J. G. Papesch, J. R. Moore, and A. E. Hawke. “Mechanical stability of pinus radiata trees at eyrewell forest investigated using static tests”. In: *New Zealand Journal of Forestry Science* 27.2 (1997), pp. 188–204.

[93] MathWorks. *ROS.org Documentation*. Accessed: 4th Nov 2013. 2013. URL: http://wiki.ros.org/.

[94] Ioan A. Şucan, Mark Moll, and Lydia E. Kavraki. “The Open Motion Planning Library”. In: *IEEE Robotics & Automation Magazine* 19.4 (Dec. 2012), pp. 72–82. DOI: 10.1109/MRA.2012.2205651.

[95] Tsuneo Yoshikawa. “Manipulability of Robotic Mechanisms”. In: *The International Journal of Robotics Research* 4.2 (1985), pp. 3–9. DOI: 10.1177/027836498500400201.

[96] L. Sciavicco and Bruno Siciliano. *Modelling and control of robot manipulators*. New York; London: Springer, 2000.

[97] Wolfram Stadler. *Analytical robotics and mechatronics*. New York: McGraw-Hill, 1995.

[98] Mark W. Spong, Seth Hutchinson, and M. Vidyasagar. *Robot modeling and control*. Hoboken, NJ: John Wiley & Sons, 2006.

[99] Peter I. Corke. *Robotics, vision and control: fundamental algorithms in MATLAB*. Vol. 73; 73. Berlin: Springer, 2011.

[100] Tsuneo Yoshikawa. “Analysis and control of robot manipulators with redundancy”. In: *1st International Symposium on Robotics Research*, pp. 735–747.

[101] Henry W. Stone. *Kinematic modeling, identification, and control of robotic manipulators*. Vol. SECS 29.; SECS 29. Kluwer Academic, 1987.

[102] Raymond A. Serway and J. W. Jewett. *Physics for Scientists and Engineers with Modern Physics*. 7th. USA: Thomson Learning Inc., 2008.

[103] Howard Anton, Irl Bivens, and Stephen Davis. *Calculus: Early Transcendentals*. 8th. New Jersey: John Wiley & Sons, 2005.

[104] Progressive Automations. *PA-17 Linear Actuator*. Accessed: 30th Apr 2015. 2015. URL: http://www.progressiveautomations.com/media/catalog/pdf/Heavy_Duty_Linear_Actuators_PA-17.pdf.

[105] AM Equipment. *Utility gearhead motors*. Accessed: 29th Apr 2015. 2015. URL: http://www.amequipment.com/ame-products/utility-gearhead-motors/.

[106] Reginald P Tewarson. *Sparse matricies*. New York: Academic Press, 1973.

[107] C.V. Meaclem et al. “K-means Partitioned Space Path Planning (KPSPP) for robotic tree felling on steep terrain”. In: *International Journal of Advanced Robotic Systems* 12.165 (2015). DOI: 10.5772/61816.

[108] Ethan Tira-Thompson. *Digital Servo Calibration and Modeling*. Report CMU-RI-TR-09-41. Robotics Institute, Carnegie Mellon University, 2009.

[109] Marc J van Kreveld and Mark H Overmars. “Divided k-d trees”. In: *Algorithmica* 6.1-6 (1991), pp. 840–858. DOI: 10.1007/BF01759075.

[110] WorkSafe New Zealand. *Best Practice Guidelines - Safe Manual Tree Felling*. Report. 2014.

[111] Kevin Lyons, John Sessions, and Jeffrey Wimer. “Effect of Undercut Style and Post Hinge Behavior in Tree Felling”. In: *Forest Science* 58.6 (2012), pp. 547–558. DOI: 10.5849/forsci.11-005.

[112] Alan Hale. *The Craft of Tree Felling*. Accessed: 22nd Jul 2014. 2008. URL: http://tnvalleywoodclub.org/articles/PDFs/Tree_Felling_Presentation.pdf.

[113] Competenz. *Best practice guidelines for Tree Felling*. 2000.

[114] Forest Industry Training and Education Council. *Best practice guidelines for Mechanised Harvesting and Processing*. Report. 2005.

[115] S. Frey, A. Dadalau, and A. Verl. “Expedient modeling of ball screw feed drives”. English. In: *Production Engineering* 6.2 (2012), pp. 205–211. DOI: 10.1007/s11740-012-0371-0.

Denavit-Hartenberg model

Referring to Chapter 4, the transformation between any two coordinate frames on the machine is given by

$$T_b^a = \prod_{i=a}^{b-1} A_{i+1}^i(q_i) \tag{1}$$

In the case of the transformation between origin and end effector (Link 8), the transform is given as

$$T_8^0 = \prod_{i=0}^{7} A_{i+1}^i(q_i) \tag{2}$$

where q_i is the joint angle for the i^{th} joint and A^i is the transformation matrix between adjacent links i and $i+1$. The transformation between any two adjacent links is given by

$$A_j^{j-1} = \begin{bmatrix} \cos(\theta_j) & -\sin(\theta_j)\cdot\cos(\alpha_j) & \sin(\theta_j)\cdot\sin(\alpha_j) & a_j\cdot\cos(\theta_j) \\ \sin(\theta_j) & \cos(\theta_j)\cdot\cos(\alpha_j) & -\cos(\theta_j)\cdot\sin(\alpha_j) & a_j\cdot\sin(\theta_j) \\ 0 & \sin(\alpha_j) & \cos(\alpha_j) & d_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

Therefore,

$$T_8^0 = A_1^0(q_1)A_2^1(q_2)A_3^2(q_3)A_4^3(q_4)A_5^4(q_5)A_6^5(q_6)A_7^6(q_7)A_8^7(q_8) \tag{4}$$

Where the transformation matrices are as such:

$$A_1^0 = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 & 0 \\ \sin(\theta_1) & \cos(\theta_1) & 0 & 0 \\ 0 & 0 & 1 & 0.296 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

$$A_2^1 = \begin{bmatrix} \cos(\theta_2) & -\sin(\theta_2) & 0 & 0.285 \\ \sin(\theta_2) & \cos(\theta_2) & 0 & 0.285\sin(\theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

$$A_3^2 = \begin{bmatrix} \cos(\theta_3) & 0 & \sin(\theta_3) & 0 \\ \sin(\theta_3) & 0 & -\cos(\theta_3) & 0.073\sin(\theta_3) \\ 0 & 1 & 0 & -0.025 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

$$A_4^3 = \begin{bmatrix} \cos(\theta_4) & -\sin(\theta_4) & 0 & 0.750 \\ \sin(\theta_4) & \cos(\theta_4) & 0 & 0.750\sin(\theta_4) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{8}$$

$$A_5^4 = \begin{bmatrix} \cos(\theta_5) & -\sin(\theta_5) & 0 & 0.750 \\ \sin(\theta_5) & \cos(\theta_5) & 0 & 0.750\sin(\theta_5) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9}$$

$$A_6^5 = \begin{bmatrix} \cos(\theta_6) & 0 & -\sin(\theta_6) & 0 \\ \sin(\theta_6) & 0 & \cos(\theta_6) & 0.073\sin(\theta_6) \\ 0 & -1 & 0 & 0.025 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

$$A_7^6 = \begin{bmatrix} \cos(\theta_7) & -\sin(\theta_7) & 0 & 0.285 \\ \sin(\theta_7) & \cos(\theta_7) & 0 & 0.285\sin(\theta_7) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{11}$$

$$A_8^7 = \begin{bmatrix} \cos(\theta_8) & -\sin(\theta_8) & 0 & 0 \\ \sin(\theta_8) & \cos(\theta_8) & 0 & 0 \\ 0 & 0 & 1 & -0.296 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{12}$$